Università degli Studi di Pisa

Dipartimento di Ingegneria dell'Informazione

Scuola di Ingegneria

Tesi di Laurea Magistrale in Ingegneria Robotica e dell'Automazione

Rilevamento di guasti sugli attuatori di un veicolo marino sovra-attuato tramite Analisi delle Componenti Principali.

Candidato:
Simone Della Tommasina

Matricola 452788

Relatore:
Prof. Andrea Caiti

Correlatori:
Ing. Simone Grechi
Ing. Filippo Fabiani

Anno Accademico 2015-2016

INDICE

ELENCO DELLE FIGURE

ELENCO DELLE TABELLE

Sommario

Questa tesi mostra come il metodo dell'Analisi delle Componenti Principali (PCA) possa essere utilizzato per risolvere il problema di Fault Detection (FD) sugli attuatori di un veicolo marino autonomo (AUV) sovra-attuato, in condizioni di guasto singolo improvviso. Dopo una breve descrizione della tecnica utilizzata, vengono presentati tutti i passaggi necessari per l'elaborazione dell'algoritmo, mettendone in luce le potenzialità e i limiti riscontrati. Si suggeriscono, inoltre, nuove soluzioni per l'isolamento dei guasti (FI) da integrare con l'algoritmo proposto. Nella parte finale della tesi vengono messi a confronto i risultati ottenuti applicando sia la PCA sia la sua variante non lineare (NLPCA). Il presente lavoro si è svolto nell'ambito del progetto V-Fides, sul quale modello dinamico sono simulati gli algoritmi.

Abstract

This thesis shows how the Principal Component Analysis (PCA) can be used to solve the problem of Fault Detection (FD) applied to over-actuated autonomous underwater vehicles (AUVs) thrusters, under single fault condition. After a short description of the method, all the necessary steps to its implementation are indicated, underscoring advantages and limitations. New solutions to Fault Isolation (FI) are proposed too. Finally, the results obtained by PCA are compared with those obtained by Nonlinear PCA (NLPCA). The proposed work is in the framework of the V-Fides project and the algorithms are demonstrated in simulation, employing the dynamical model of that vehicle.

INTRODUZIONE

Nel campo della robotica, come in numerosi altri settori, la realizzazione di sistemi automattizzati più affidabili e robusti rappresenta uno degli obiettivi di maggiore interesse. Per questo motivo può essere di grande utilità lo sviluppo di algoritmi in grado sia di rilevare la presenza di un guasto (FD) sia di riconoscerne il tipo (FI).
L'importanza di questo aspetto è evidente se consideriamo il caso dei veicoli marini autonomi sovra-attuati. Grazie alla sovra-attuazione, infatti, il veicolo potrebbe riuscire a mantenersi stabile e, in certi casi, persino a portare a termine la missione, anche quando uno dei suoi thruster è fuori uso. Per farlo però, è necessario che il sistema di controllo che determina i comandi agli attuatori del veicolo, venga informato del guasto, così da poter escludere il thruster danneggiato e ottimizzare l'azione di quelli ancora funzionanti.

In letteratura sono presenti numerosi metodi per affrontare il problema della rilevazione e isolamento di guasti e uno di questi è l'Analisi delle Componente Principali (PCA), ovvero una tecnica ampiamente utilizzata nello studio e monitoraggio di sistemi complessi. La PCA trova grande applicazione nell'ambito degli impianti chimici, nel quale si è dimostrata essere in grado di fornire validi risultati, come discusso in [12–15].

L'obiettivo principale di questa tesi è proprio quello di valutare l'applicabilità della PCA alla risoluzione del problema di rilevamento dei guasti sugli attuatori di un veicolo marino autonomo sovra-attuato.

Il primo capitolo introduce la tecnica della PCA esponendone i concetti chiave e fornendone una trattazione analitica.

Nel secondo capitolo viene data una descrizione del veicolo subacqueo V-Fides e il modello matematico corrispondente utilizzato per testare gli algoritmi sviluppati.

Il terzo capitolo spiega come poter applicare la PCA su un veicolo subacqueo, riportando i risultati ottenuti attraverso opportune simulazioni.

Il quarto capitolo presenta una variante non lineare della PCA mostrando le potenzialità e i limiti di una soluzione di questo tipo.

Nel quinto capitolo, infine, vengono messe a confronto le prestazioni delle due tecniche sopraindicate.

CAPITOLO 1

TECNICA DELLA PCA

L'**Analisi delle Componenti Principali** (PCA) è una tecnica ampiamente utilizzata nel monitoraggio di sistemi complessi e nella rilevazione ed individuazione di possibili guasti. Attraverso un'opportuna trasformazione lineare, la PCA permette di ridurre la dimensione del sistema in esame mantenendo comunque il maggior contenuto informativo possibile. Per analizzare il comportamento di un sistema non sarà necessario quindi studiare l'andamento nel tempo di tutte le variabili di interesse, ma sarà sufficiente prendere in considerazione solo quello di un nuovo insieme di variabili più piccolo del precedente e di cui ogni elemento risulta una combinazione lineare delle variabili originali.
Nell'approccio classico, con prima componente principale si intende la direzione lungo la quale le osservazioni del sistema (in condizioni operative e in assenza di guasti) presentano varianza massima. Analogamente la seconda componente sarà la direzione ortogonale alla precedente che massimizza la varianza dei dati proiettati su di essa. Procedendo in questo modo possono essere così definite tutte le componenti principali di un sistema. Attraverso la PCA è quindi possibile descrivere le relazioni tra le variabili ed individuare la presenza di eventuali comportamenti anomali pur non disponendo di un modello matematico del sistema. Il fatto che la PCA sia completamente model-free, rappresenta uno degli aspetti più interessanti di questa tecnica.
Questo la rende particolarmente adatta nello studio di sistemi complessi come molti impianti chimici in cui spesso trovare un modello matematico risulta un'operazione tutt'altro che semplice. Anche quando siano note le relazioni fra le variabili del sistema, come nel caso di alcuni veicoli o sistemi meccanici, può comunque essere conveniente utilizzare una tecnica model-free data l'inevitabile presenza di dinamiche non modellate e parametri difficilmente stimabili in modo esatto.

Nel primo paragrafo di questo capitolo verranno presentati la PCA e i concetti necessari alla sua comprensione; successivamente verrà mostrato come tale tecnica, nella sua formulazione standard, possa essere utilizzata per risolvere i problemi di **Fault Detection e Fault Isolation** (FDI).

1.1 Componenti Principali

Supponiamo di avere una matrice dei dati $X \in \mathbb{R}^{N\times n}$, contenente N misure delle n variabili del sistema e indichiamo con $x_i^T \in \mathbb{R}^n$ l'i-esima riga della matrice X . È opportuno osservare che tra le variabili del sistema sono presenti tutti i segnali di interesse che possono essere misurati o stimati. Non vi è quindi alcuna distinzione tra ingressi e uscite che andranno entrambi a costituire il vettore dei dati x_i.

Detta $\Sigma \in \mathbb{R}^{n\times n}$ la matrice di covarianza di X e $l < n$ il numero delle componenti principali del sistema, chiamiamo **PCA Loading Matrix** la matrice $P \in \mathbb{R}^{n\times l}$ le cui colonne altro non sono che gli autovettori di Σ associati ai suoi l autovalori più grandi. Tali autovettori rappresentano proprio le componenti principali del sistema considerato.
Uno degli obiettivi della PCA è quello di diminuire la dimensione del sistema attraverso una trasformazione lineare ottima che riduca al minimo la perdita di informazione. Possiamo, a questo

proposito, definire un vettore $t_i \in \mathbb{R}^l$, detto *score vector*, tale che

$$t_i = P^T x_i \quad . \tag{1.1}$$

Lo *score vector* è, come desiderato, un vettore a dimensione ridotta in grado di catturare le relazioni tra le variabili originali del sistema.

1.2 Fault Detection

Una volta determinata la matrice P, le cui colonne, come già affermato precedentemente, rappresentano le componenti principali del sistema, possiamo definire due sottospazi di interesse:

- *Sottospazio delle Componenti Principali* (**PCS**) $\rightarrow$ Sottospazio di dimensione l contenente le direzioni di massima correlazione dei dati
- *Sottospazio Residuo* (**RS**) $\rightarrow$ Sottospazio di dimensione $n-l$, ortogonale al PCS.

Il primo rappresenta un modello delle relazioni tra le variabili del sistema misurate in condizioni operative, mentre il secondo ci fornisce un'indicazione di quanto queste si discostino dal loro comportamento normale.

A questo punto, come mostrato in [4], possiamo scomporre $x_i \in \mathbb{R}^n$, vettore dei dati acquisito all'istante i, in due componenti

$$x_i = \hat{x}_i + \tilde{x}_i \tag{1.2}$$

dove $\hat{x}_i$ rappresenta la proiezione di x_i su PCS, mentre $\tilde{x}_i$ indica quella su RS.
Il vettore $\hat{x}_i$, calcolabile come

$$\hat{x}_i = Pt_i = PP^T x_i = Cx_i \quad , \tag{1.3}$$

indica la componente di x_i prevista dal modello. La componente residua

$$\tilde{x}_i = (I_n - C)x_i = \tilde{C}x_i \tag{1.4}$$

dove $I_n \in \mathbb{R}^{n\times n}$ è la matrice identità di dimensione n, ci fornisce quindi un'indicazione di quanto la misura acquisita si "allontani" da quella modellata. $C, \tilde{C} \in \mathbb{R}^{n\times n}$ rappresentano, rispettivamente, la mtrice di proiezione su PCS e su RS. In condizioni operative ideali, ovvero in assenza di disturbi o guasti, le relazioni tra le variabili vengono mantenute, questo significa che il sistema evolve muovendosi solo all'interno del PCS. In tal caso, perciò, la componente residua risulta nulla o molto piccola. La presenza di una componente residua non trascurabile, invece, è sintomo che qualcosa non sta funzionando come ci si aspetterebbe e che quindi siamo di fronte ad una situazione anomala. Questo ci suggerisce una strategia per affrontare e risolvere il problema di Fault Detection (FD). Dopo aver elaborato un modello PCA per il sistema in esame sulla base di un dataset opportuno, è sufficiente monitorare on-line la componente residua delle variabili misurate e assicurarci che questa non raggiunga valori troppo "elevati" rispetto a quelli ottenuti in condizioni nominali. Dal momento che $\tilde{x}_i$ è un vettore, non è sempre facile stabilire se il valore attuale è anomalo oppure no, per questo motivo può essere più conveniente ricorrere ad un indice scalare detto *squared prediction error* (SPE):

$$SPE = \|\tilde{x}_i\|^2 = \left\|\tilde{C}x_i\right\|^2 \quad . \tag{1.5}$$

In conclusione avremo che un guasto è rilevato ogni qual volta SPE raggiunga un valore sopra una soglia prefissata basata sul comportamento del sistema in condizioni nominali. La determinazione della soglia rappresenta un punto critico della tecnica, infatti non deve essere troppo piccola, in modo da evitare così la generazione di falsi allarmi dovuti a rumore o a normali fluttuazioni nel sistema, ma non deve essere nemmeno troppo grande da rischiare di non riconoscere dei guasti.

1.3 Fault Isolation

Nel paragrafo precedente si è mostrato come la PCA possa essere utilizzata per individuare un'anomalia nel sistema, ma non è stata data nessuna indicazione su come identificare il guasto che ha provocato tale comportamento. Questo aspetto viene analizzato in [10] dove il problema di Fault Isolation (FI) è affrontato con il metodo della ricostruzione. Si ipotizza, innanzitutto, che ogni guasto singolo possa influenzare una sola variabile. Perciò se un guasto va ad influire sul comportamento di r variabili, allora questo sarà modellato come un guasto r-dimensionale. Detta quindi r la dimensione di un guasto, a questo saranno associati un insieme R contenente gli indici delle variabili coinvolte e una matrice Ξ_R di dimensione $n \times r$. L' i-esima colonna di Ξ_R avrà tutti 0 tranne un unico 1 nella posizione indicata dall' i-esimo elemento di R. Consideriamo, per esempio, un sistema di dimensione $n = 5$ con un guasto di dimensione $r = 2$ e supponiamo $R = \{2,4\}$. La matrice Ξ_R in questo caso sarà:

$$\Xi_R = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}^T \tag{1.6}$$

La matrice Ξ_R gioca un ruolo fondamentale per la FI, infatti presenta le direzioni di ricostruzione associate al guasto R. Con ricostruzione delle variabili del sistema, si intende la stima del vettore ricostruito $\hat{x}_R$, ottenuto eliminando gli effetti del guasto R. Il vettore ricostruito può essere calcolato come

$$\hat{x}_R = G_R x \tag{1.7}$$

dove

$$\tilde{\Xi}_R = \tilde{C}\Xi_R \tag{1.8}$$

$$G_R = I_n - \Xi_R(\tilde{\Xi}_R^T\tilde{\Xi}_R)^{-1}\tilde{\Xi}_R^T \ . \tag{1.9}$$

L'idea è concettualmente molto semplice: supponiamo che il modulo FD segnali la presenza di un guasto; per capire di quale guasto si tratta, ne ipotizziamo uno qualsiasi tra quelli possibili e cerchiamo di eliminarne gli effetti tramite la procedura di ricostruzione. A questo punto se il vettore ricostruito giace nel PCS (ovvero la sua componente residua $\tilde{x}_R$ è piccola) allora significa che il guasto selezionato è proprio quello realmente avvenuto nel sistema e l'algoritmo si interrompe. In caso contrario questo viene scartato e si applica la stessa procedura con un nuovo candidato. Come nel caso FD è più conveniente servirsi di un indice scalare rispetto ad uno vettoriale, per questo motivo si procede con il calcolo dei **residui strutturati** SPE_R:

$$\tilde{x}_R = \tilde{C}\hat{x}_R = P_R x \tag{1.10}$$

$$P_R = \tilde{C}G_R = \tilde{C} - \tilde{\Xi}_R(\tilde{\Xi}_R^T\tilde{\Xi}_R)^{-1}\tilde{\Xi}_R^T \ . \tag{1.11}$$

$$SPE_R = \|\tilde{x}_R\|^2 \ . \tag{1.12}$$

Ogni guasto è associato ad un residuo specifico il cui comportamento, monitorabile *on-line*, ci permette di stabilire se tale guasto si è verificato oppure no. È evidente che per comprendere se un residuo stia segnalando o meno la presenza di un'anomalia, è necessario fissare un set di soglie tarate opportunamente.

1.4 Fault Detectability e Isolatability

Come già affermato precedentemente, l'idea alla base della PCA è che in condizioni ideali (assenza di guasti o disturbi) la normale correlazione tra le variabili del sistema viene mantenuta, mentre l'insorgere di un guasto porta necessariamente ad un suo cambiamento. Questo però non è sempre detto, infatti è possibile che l'azione di un guasto non vada a modificare la struttura di correlazione dei dati lasciando il valore di SPE sotto soglia. Se tale situazione si verificasse, allora il guasto non sarebbe rilevabile con questa tecnica e per la sua individuazione diventerebbe necessario procedere attraverso altre vie. Inoltre è possibile anche che due o più guasti influenzino la correlazione tra le misure in modo analogo, impedendo alla PCA di poter distinguere gli uni dagli altri. Diamo a questo punto una descrizione formale dei concetti appena espressi.

Consideriamo un generico guasto R di dimensione r e supponiamo che questo sia di tipo additivo, ovvero

$$x_i = x_i^0 + \Xi_R f \tag{1.13}$$

dove x_i, $x_i^0 \in \mathbb{R}^n$ indicano rispettivamente la misura acquisita all'istante i e il vettore ideale non perturbato dal guasto, mentre $f \in \mathbb{R}^r$ è il vettore di fault. Proiettando x_i su RS e trascurando la componente residua dovuta a x_i^0 (dal momento che questo rappresenta la misura ideale), avremo che

$$\tilde{x}_i = \tilde{C} x_i = \tilde{\Xi}_R f \quad . \tag{1.14}$$

È evidente che il guasto è rilevabile se e soltanto se $\tilde{\Xi}_R f \neq 0$, ovvero quando $f \notin ker\{\tilde{\Xi}_R\}$. Se ciò non succede significa che il guasto non ha modificato le relazioni tra le variabili del sistema ma la sua azione è rimasta confinata all'interno del PCS. Per assicurarci che un tipo di guasto possa sempre essere rilevabile, è necessario che $ker\{\tilde{\Xi}_R\} = \{0\}$.

Affinché due guasti A e B siano sempre distinguibili, è richiesto che questi siano in grado di modificare la struttura di correlazione dei dati in modo differente. La condizione che garantisce perciò la completa isolabilità di A e B sarà

$$span\{\tilde{\Xi}_A\} \cap span\{\tilde{\Xi}_B\} = \emptyset \quad . \tag{1.15}$$

In [10] è presentato un algoritmo che permette di studiare in modo semplice la rilevabilità e l'isolabilità di un insieme di possibili guasti.
Tale algoritmo sfrutta l'operatore *Rcond* definito come il rapporto tra il minimo ed il massimo valore singolare di una matrice:

$$Rcond(\tilde{\Xi}_R) = \frac{\sigma_{min}(\tilde{\Xi}_R)}{\sigma_{max}(\tilde{\Xi}_R)} \quad . \tag{1.16}$$

Algoritmo per Fault Detectability e Fault Isolatability:

1. $r = 1$ (single-fault) $\rightarrow$ Calcolare $\tilde{\Xi}_R$ per tutti i guasti. Se $\tilde{\Xi}_R^T \tilde{\Xi}_R \simeq 0$, il fault non è rilevabile e la direzione indicata da R non è presa in considerazione
2. $r = r + 1 \rightarrow$ Per tutti gli insiemi R di direzioni non ancora scartate calcolare $Rcond(\tilde{\Xi}_R)$:
   - Se $Rcond(\tilde{\Xi}_R) \simeq 0 \rightarrow$ un fault simultaneo su queste r variabili non è rilevabile (questa combinazione va scartata). Tutte le combinazioni di $r-1$ variabili tra quelle considerate non sono isolabili tra loro
   - Se $Rcond(\tilde{\Xi}_R) \gg 0 \rightarrow$ tutte le combinazioni di $r-1$ variabili tra quelle selezionate sono isolabili
3. Se $r \leq n - l$ torna al passo 2.

1.5 Numero componenti principali

Finora abbiamo supposto di conoscere il numero l di componenti principali necessarie per modellare un sistema, ma in realtà determinare tale numero è un problema tutt'altro che banale. Se l è troppo piccolo, infatti, il modello non è in grado di "catturare" tutte le informazioni fornite dai dati ottenendo così una rappresentazione non accurata del sistema. D'altra parte scegliere un valore di l troppo elevato porta ad una "sovra-parametrizzazione" dei dati, andando così ad includere nel modello anche gli effetti dovuti al rumore sulle misure.
In [11] sono presentati alcuni criteri utilizzati per stabilire il numero corretto di componenti principali. Tali criteri prevedono la definizioni di indici che suggeriscono quale valore di l sia più opprtuno scegliere. Di seguito sono riportati alcuni dei metodi proposti.

- *Akaike Information Criterion* (AIC) $\rightarrow$ questo metodo suggerisce di scegliere l in modo da massimizzare l'indice $AIC(l)$ definito come

$$AIC(l) = -2\log\left(\frac{\prod_{i=l+1}^{n}\lambda_i^{\frac{1}{n-l}}}{\frac{1}{n-l}\sum_{i=l+1}^{n}\lambda_i}\right)^{N(n-l)} + l(n-l+1) \tag{1.17}$$

  dove λ_i indica l'i-esimo autovalore della matrice di covarianza Σ in ordine decrescente e N è il numero delle misure.

- *Minimum Description Length* (MDL) $\rightarrow$ anche in questo caso l'obiettivo è quello di trovare l in modo da massimizzare l'indice

$$MDL(l) = -2\log\left(\frac{\prod_{i=l+1}^{n}\lambda_i^{\frac{1}{n-l}}}{\frac{1}{n-l}\sum_{i=l+1}^{n}\lambda_i}\right)^{N(n-l)} + \frac{1}{2}l(n-l+1)\log N \tag{1.18}$$

I risultati, data l'analogia degli indici, sono molto simili a quelli ottenuti con il metodo AIC. È opportuno osservare che nel caso in cui non esista un punto di massimo per le funzioni in 1.17 e 1.18, allora i due criteri non sono in grado di fornirci alcun supporto nella determinazione di l.

- *Imbedded Error Function* (IEF) $\rightarrow$ In questo caso la scelta di l ricade sul punto di minimo della funzione

$$IEF(l) = \left(\frac{l \sum_{i=l+1}^{n} \lambda_i}{Nn(n-l)} \right)^{\frac{1}{2}} \quad . \tag{1.19}$$

  Come per i criteri precedenti, è possibile che il punto ottimo cercato non esista. In questo caso il metodo risulta inefficace.

- *Cumulative Percent Variance* (CPV) $\rightarrow$ questo metodo fornisce una misura della percentuale di varianza catturata dai primi l autovalori di Σ. Per poter scegliere il numero di componenti principali sarà necessario quindi fissare una soglia (es: 90%, 95%, 99% ecc ...) e determinare per quale valore di l tale soglia viene superata.

$$CPV(l) = 100 \left(\frac{\sum_{i=1}^{l} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \right) \% \quad . \tag{1.20}$$

  Questo criterio, rispetto ai precedenti, risulta meno "oggettivo" dal momento che la soglia è fissata in modo assolutamente arbitrario.

- *Residual Percent Variance* (RPV) $\rightarrow$ a differenza del metodo precedente, in questo caso si valuta quanta percentuale di varianza è fornita dagli autovalori residui, ovvero quelli scartati a seconda della scelta di l. In questo caso ciò che viene cercato è il "ginocchio" della funzione

$$RPV(l) = 100 \left(\frac{\sum_{i=l+1}^{n} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \right) \% \quad . \tag{1.21}$$

  Il problema principale di questo metodo è che non sempre è facile identificare un ginocchio nella funzione 1.21.

- *Parallel Analysis* (PA) $\rightarrow$ L'idea è quella di considerare solo gli autovalori che vanno effettivamente a contribuire alla correlazione dei dati misurati. Per questo motivo, oltre al dataset originale, andremo a considerare anche un nuovo dataset della stessa dimensione, costituito da variabili non correlate tra di loro. A questo punto verranno tracciati sullo stesso grafico gli autovalori di entrambe le matrici di correlazione. Gli autovalori che si troveranno sopra il punto di intersezione tra le due curve, sono quelli corrispondenti alle componenti principali del sistema.

1.6 FDI con PCA

Riepiloghiamo a questo punto i passi necessari per l'applicazione della PCA al problema di Fault Detection e Isolation.

Algoritmo per Fault Detection e Isolation con Principal Component Analysis:

1. Acquisire un dataset X, costituito dalle variabili di interesse del sistema, misurate in condizioni operative.

   OFF-LINE:

2. Calcolare la matrice di covarianza Σ di X.
3. Determinare il numero l di componenti principali (sfruttando uno o più algoritmi tra quelli presentati nel paragrafo 1.5) e la Loading Matrix P.
4. Calcolare la matrice $\tilde{C}$ di proiezione su RS.
5. Predisporre l'insieme $\mathfrak{F}$ dei fault possibili e determinare per ognuno la matrice Ξ_R corrispondente.
6. Applicare l'algoritmo presentato nel paragrafo 1.4 per studiare l'isolabilità dei guasti considerati. Eliminare da $\mathfrak{F}$ i guasti che non possono essere rilevati.
7. Calcolare per ogni $R \in \mathfrak{F}$ la matrice P_R (per il calcolo del residuo strutturato associato).
8. Fissare una soglia $\theta_R \ \forall \ R \in \mathfrak{F}$.

   ON-LINE:

9. Misurare, ad ogni istante di campionamento t, il vettore $x(t)$ costituito dalle variabili del sistema.
10. Calcolare $SPE_R = \|P_R x(t)\|^2 \ \forall \ R \in \mathfrak{F}$:
    - Se $SPE_R < \theta_R \ \forall \ R \in \mathfrak{F} \rightarrow$ **NO FAULT**
    - Se $\exists! \ R \mid SPE_R < \theta_R \ \rightarrow$ **FAULT R**
11. Tornare al passo 9.

I vantaggi offerti da questa tecnica, rispetto ai metodi standard utilizzati per FDI, sono molteplici. Innanzitutto, come già affermato precedentemente, può essere applicata anche quando non si ha a disposizione un modello del sistema, rendendola adatta ad una grande varietà di situazioni. La PCA, inoltre, non presenta nemmeno un elevato carico computazionale, dato che non richiede operazioni come inversione on-line di matrici o calcolo di integrali, ma riesce comunque a gestire casi di fault multi-dimensionale, permettendo quindi di individuare e isolare anche condizioni di guasti multipli.
È bene però, tener conto anche dei limiti dovuti all'applicazione di questa tecnica. Un primo problema è quello dovuto alla modellazione del sistema che avviene tramite l'acquisizione di un dataset opportuno. I dati utilizzati, infatti, saranno necessariamente affetti da rumore e outliers e questo può portare ad un'errata rappresentazione del PCS. Inoltre è possibile che dei semplici cambiamenti nelle condizioni operative, dovuti ad esempio a fattori ambientali, siano riconosciuti dal modulo di FDI come situazioni anomale, portando così alla generazione di falsi allarmi. È

quindi fondamentale tarare opportunamente le soglie presenti e scegliere correttamente il dataset con cui "addestrare" l'algoritmo. Non va dimenticato, inoltre, il problema dovuto alla possibile presenza di guasti non in grado di modificare la struttura di correlazione delle variabili del sistema. Questi fault, come già discusso nel paragrafo 1.4, non possono in alcun modo essere rilevati tramite PCA. Infine è opportuno osservare un ulteriore aspetto che potrebbe portare a dei problemi nell'utilizzo di questo metodo. Si è sempre supposto che i guasti singoli vadano ad influenzare un'unica variabile del sistema, lasciando però immutata la correlazione tra le altre. Questo metodo risulta particolarmente efficiente nell'individuazione di eventuali fault sui sensori, ma potrebbe essere inadeguato per trattare quelli su alcuni attuatori. Un'azione anomala da parte di quest'ultimi, potrebbe infatti andare a modificare il comportamento di tutte le variabili in gioco, rendendo difficile capire come procedere con la fase di ricostruzione del vettore dei dati x. Questo aspetto può risultare particolarmente critico quando il sistema è soggetto a vari anelli di controllo.

CAPITOLO 2

DESCRIZIONE V-FIDES AUV

L'obiettivo di questo capitolo è quello di fornire una descrizione del sistema per il quale intendiamo realizzare un modulo di FDI e presentare brevemente il modello (sviluppato in Simulink) utilizzato per testarne la validità. Per una trattazione più approfondita dell'argomento, fare riferimento al lavoro presentato in [1] e [5].
Il sistema considerato si chiama V-Fides ed è un veicolo subacqueo general purpose sviluppato da WASS S.p.A (Whitehead Sistemi Subacquei, Livorno) con la partecipazione di altri partners tra cui l'Università di Pisa. Tale veicolo, pensato per operazioni di esplorazione e monitoraggio di fondale a profondità elevate (fino a 3000 [m]), può funzionare sia in modo autonomo (AUV), che controllato da remoto (ROV) ed è dotato di 7 thrusters con caratteristica ingresso-uscita asimmetrica. La sovrattuazione del sistema gli permette anche in caso di guasti, di riuscire a portare a termine alcuni task, seppure con prestazioni ridotte. Un sistema di controllo che integra le misure dei sensori con le informazioni ottenute tramite un algoritmo di FDI, può quindi modificare la sua azione ottimizzandola a seconda dello stato degli attuatori.
V-Fides è dotato, inoltre, di un payload per la navigazione autonoma con Inertial Measurement Unit (IMU), Doppler Velocity Logger (DVL), sensore di profondità, bussola magnetica e un modem acustico per la localizzazione. In figura 2.1 sono mostrate due immagini del veicolo, con numerazione dei vari thruster.

2.1 Modello del Veicolo

Le equazioni che descrivono la dinamica del sistema, sono quelle di un veicolo subacqueo autonomo a 6 gradi di libertà. In accordo a quanto presentato in [6] avremo

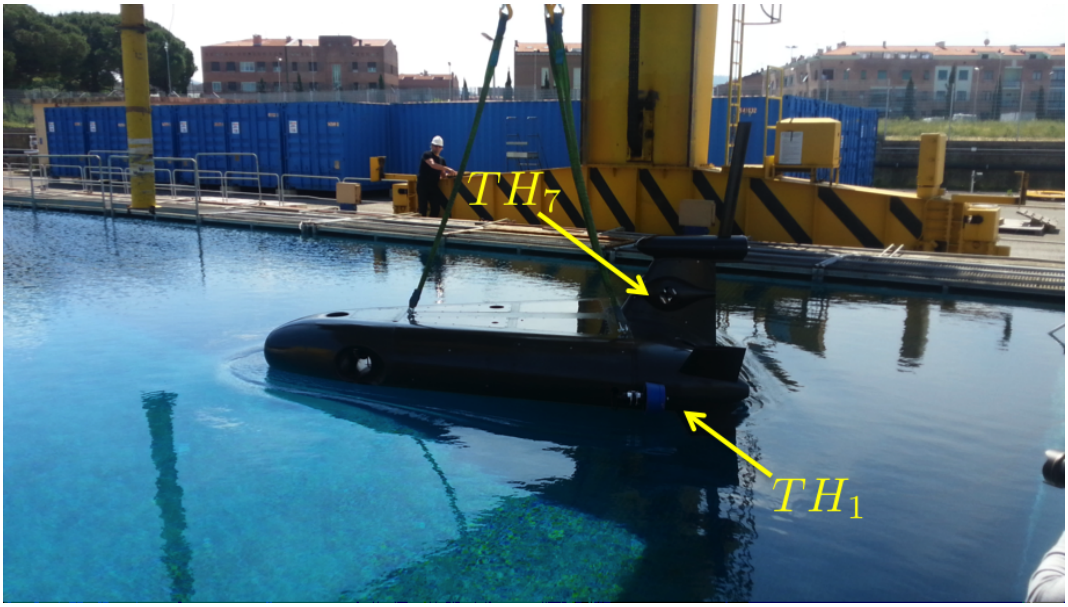
$$\begin{aligned} &\dot{\eta} = J(\eta)\nu \\ &M\dot{\nu} + C(\nu)\nu + D(\nu)\nu + g(\eta) = \tau \end{aligned} \tag{2.1}$$

dove $\eta = [x, y, z, \phi, \theta, \psi]^T$ è il vettore delle posizioni generalizzate in *navigation frame* (NED) e angoli di Eulero, mentre $\nu = [u, v, w, p, q, r]^T$ è il vettore delle velocità lineari ed angolari in *body frame*. La prima equazione descrive il modello cinematico dell'AUV e $J(\cdot)$ ne rappresenta la matrice Jacobiana. La seconda equazione, invece, presenta la dinamica del sistema in forma lagrangiana e i termini M,$C(\nu)$,$D(\nu)$ e $g(\eta)$ rappresentano rispettivamente la matrice delle inerzie e delle masse aggiunte, la matrice di Coriolis, la matrice di attrito idrodinamico e il vettore delle forze di galleggiamento (buoyancy e gravità). Con τ è stato indicato, infine, il vettore di forze e momenti generalizzati (*wrench* in *body frame*).

2.2 Sistemi di Controllo

Nel veicolo possono essere individuati tre anelli principali di controllo tra loro interconnessi: sistema di guida, autopilota e sistema di allocazione ottima del controllo.
Il primo prevede una guida basata su waypoint e implementa una legge simile alla classica *Proportional Navigation Guidance* (PNG). Le uscite del sitema sono le velocità desiderate di avanza-

(a)



(b)

Figura 2.1: Due viste di V-Fides, con numerazione dei vari thrusters.

mento (*surge -yaw*) e verticale (*heave*). È opportuno osservare che la dinamica del veicolo risulta parzialmente linearizzata e disaccopiata tra questi due moti.
L'autopitola, invece, si occupa di determinare il vettore delle forze e momenti generalizzati τ_d, che permette al veicolo di inseguire le velocità di riferimento. Questo è a sua volta costituito da due anelli di controllo. Il primo regola a zero l'assetto $\Theta = [\phi, \theta]^T$ (*roll* e *pitch*) e la sua velocità $\dot{\Theta}$

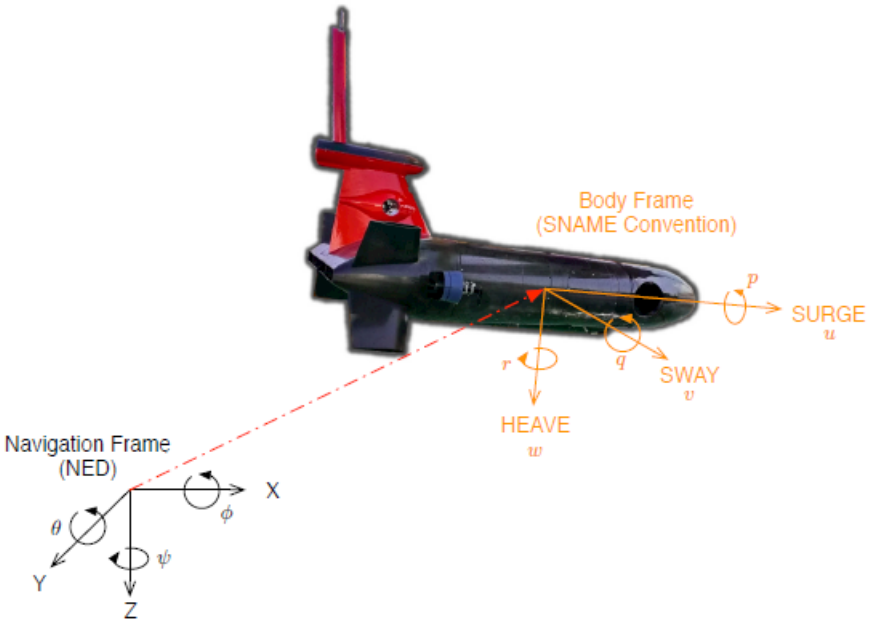


Figura 2.2: Sistemi di riferimento su AUV.

tramite un controllore PID:

$$\tau_\Theta = -K_{D\Theta}\dot{\Theta} - K_{P\Theta}\Theta - K_{I_\Theta}\int_t \Theta\, dt \quad . \tag{2.2}$$

Inoltre, al fine di ridurre i gradi di libertà del sistema da 6 a 3, la velocità v di *sway* viene forzata a zero attraverso un controllo PI:

$$\tau_y = -K_{Py}v - K_{Iy}\int_t v\, dt \quad . \tag{2.3}$$

Il secondo anello di controllo, invece, impiegando 3 controllori PI, calcola le forze necessarie per l'inseguimento della legge cinematica imposta dal sistema di guida:

$$\tau_x = K_{Px}(u_d - u) + K_{Ix}\int_t (u_d - u)\, dt \tag{2.4}$$

$$\tau_z = K_{Pz}(w_d - w) + K_{Iz}\int_t (w_d - w)\, dt \tag{2.5}$$

$$\tau_\psi = K_{P\psi}(r_d - r) + K_{I\psi}\int_t (r_d - r)\, dt \quad . \tag{2.6}$$

Il vettore delle forze così ottenuto, è espresso in *body frame* e ha dimensione 6. Dal momento che V-Fides è sovra-attuato e dispone di 7 thruster, è necessario effettuare una mappatura tra τ_d e i comadi da fornire in ingresso a ciascun attuatore ($u_i \in U$). Tale operazione è svolta dal sistema di allocazione ottima del controllo.
Indicando con $F_i^b, M_i^b \in \mathbb{R}^3$ rispettivamente la forza ([N]) e il momento ([Nm]) espressi in *body frame*, esercitati dall'i-esimo attuatore, avremo che:

$$F_i^b = a_i k_i(u_i) u_i \tag{2.7}$$

$$M_i^b = p_i^b \wedge F_i^b \tag{2.8}$$

dove:

- $k_i(\cdot) \in \mathbb{R} \rightarrow$ guadagno dimensionale in [N] su singolo thruster che modella la caratteristica asimmetrica del'attuatore (figura 2.3) a seconda del verso di rotazione dell'elica;
- $a_i \in \mathbb{R}^3 \rightarrow$ asse lungo la quale è prodotta la forza (versore);
- $u_i \in U = \{u \in \mathbb{R} \mid |u| \leq 1\} \rightarrow$ ingresso effettivo dell'attuatore (adimensionale);
- $p_i^b \in \mathbb{R}^3 \rightarrow$ posizione del thruster i-esimo rispetto al centro di massa del veicolo.

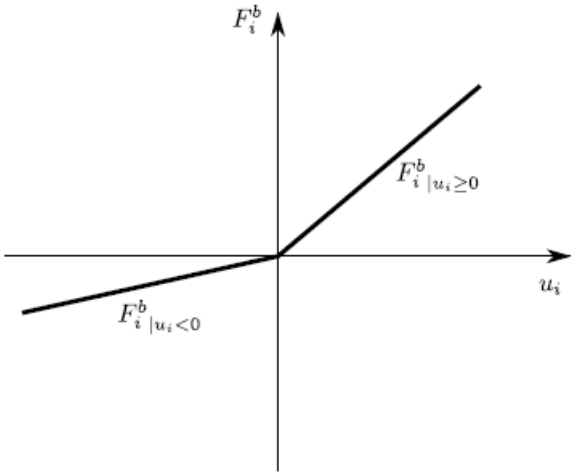


Figura 2.3: Rappresentazione caratteristica asimmetrica del thruster *i*.

La risultante delle forze sarà

$$\sum_{i=1}^{7} \begin{bmatrix} F_i^b \\ M_i^b \end{bmatrix} = \tau = B(p,a)Ku \tag{2.9}$$

con $B \in \mathbb{R}^{6\times 7}$ e $K = diag(K_i) \in \mathbb{R}^{7\times 7}$.
Noto $\tau_d = [\tau_{dl}^T, \tau_{da}^T]^T$, dove $\tau_{dl}, \tau_{da} \in \mathbb{R}^3$ sono rispettivamente la parte del vettore di forze e coppie generalizzate richieste dal controllo, il vettore $u \in \mathbb{R}^7$, costituito dai comandi ai thruster, sarà ottenuto mediante un processo di ottimizzazione di tipo *Linear Programming* (LP). In accordo allo schema mostrato in figura 2.4, il problema sarà formulato nel modo seguente:

$$\begin{aligned} \min_{\alpha_s,\alpha_f,\alpha_l,\alpha_a} \; & K_s\alpha_s + K_f\alpha_f + K_l\alpha_l + K_a\alpha_a \\ \text{subject to } & -\alpha_s \leq s \leq \alpha_s \\ & -\alpha_f \leq q_f \leq \alpha_f \\ & -\alpha_l \leq q_l \leq \alpha_l \\ & -\alpha_a \leq q_a \leq \alpha_a \\ & f_{min} \leq f \leq f_{max} \\ & \alpha_s, \alpha_f, \alpha_l, \alpha_a \geq 0 \end{aligned} \tag{2.10}$$

dove:

- $f_i = k_i(u_i)u_i \rightarrow i$-esima forza allocata;
- $s = \tau_d - B(p,a)f \rightarrow$ errore di allocazione del controllo;

- $q_l = (B(p,a)f)_l \wedge \tau_{dl} \rightarrow$ disallineamento lineare;
- $q_a = (B(p,a)f)_a \wedge \tau_{da} \rightarrow$ disallineamento angolare;
- $q_l = [\ |f_1|\ , \ldots\ , |f_7|\ ]^T \rightarrow$ vettore dei moduli delle forze allocate;
- $K_s \in \mathbb{R}_+^{1\times 6}, K_l \in \mathbb{R}_+^{1\times 3}, K_a \in \mathbb{R}_+^{1\times 3}, K_f \in \mathbb{R}_+^{1\times 7} \rightarrow$ matrici di peso.

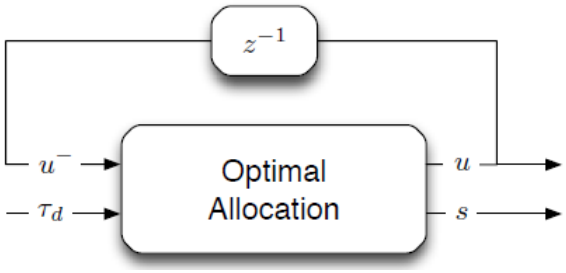


Figura 2.4: Schema del sistema di allocazione ottimo del controllo.

2.3 Modello Simulink

In figura 2.5, è mostrata un'immagine del modello Simulink utilizzato per studiare la dinamica del veicolo controllato e verificare l'efficacia degli algoritmi di FDI sviluppati.

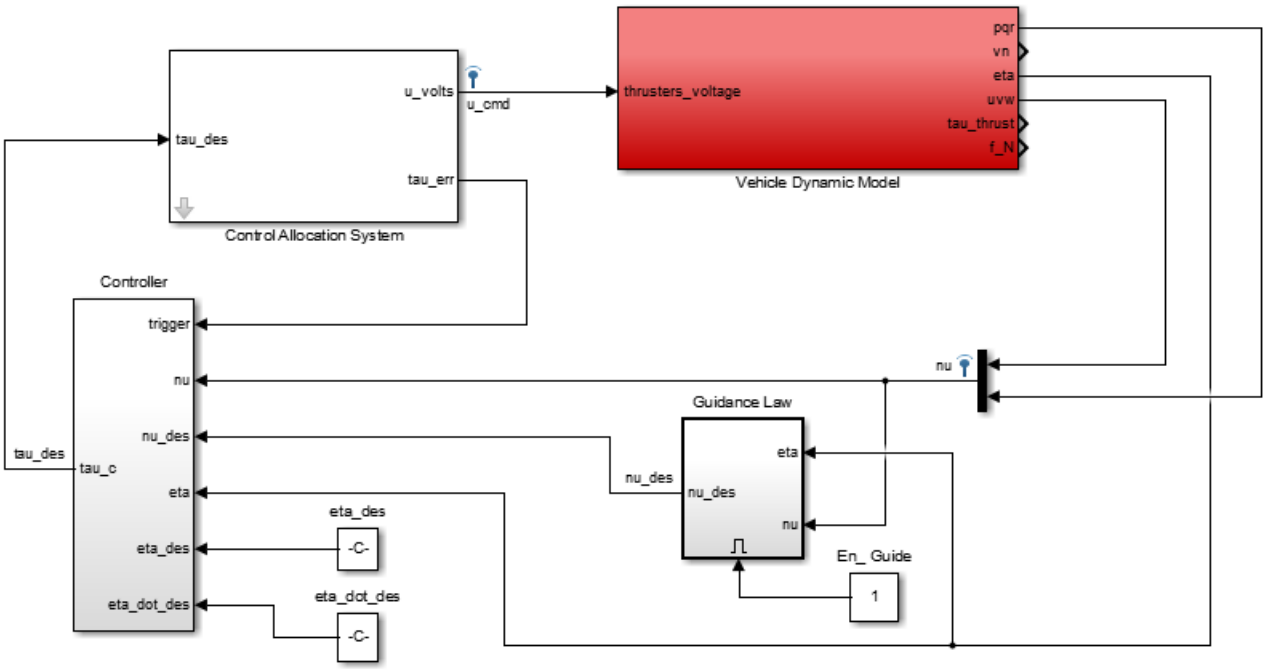


Figura 2.5: Schema Simulink di V-Fides.

Si può osservare la presenza di quattro blocchi principali:

- Vehicle Dynamic Model → questo blocco simula la dinamica del sistema, prendendo in ingresso i comandi agli attuatori e restituendo in uscita i vettori η e $\nu = [u\ v\ w\ p\ q\ r]^T$. In uscita dal blocco troviamo anche il vettore f_N e τ_{thrust}, contenenti rispettivamente i moduli delle forze esercitate dai thruster e le forze e i momenti generalizzati espressi in *body frame*. Questi ultimi due possono essere valutati solo in fase di simulazione. Nella pratica, infatti, non è previsto alcun sensore sul veicolo, che permetta di misurare le forze esercitate dagli attuatori. All'interno di questo blocco è simulata la presenza di eventuali guasti, ottenuta annullando l'azione del thruster rotto.

- Guidance Law → questo blocco implementa un sistema di guida utilizzato per permettere al veicolo di eseguire una tipica survey. Prendendo in ingresso le uscite del blocco precedente, il sistema calcola la ν desiderata da mandare in ingresso al controllore.

- Controller → questo blocco rappresenta l'autopilota ed è utilizzato per determinare la τ che si desidera applicare al veicolo.

- Control Allocation System → il sistema di allocazione del controllo, utilizzato durante le simulazioni, è quello descritto nel paragrafo precedente. Attraverso una procedura di ottimizzazione *Linear Programming*, vengono calcolati i comandi effettivamenente inviati ai thruster.

CAPITOLO 3

PCA APPLICATA A V-FIDES

Nel capitolo 1 è stata introdotta la tecnica dell' Analisi delle Componenti Principali (PCA) ed è stato mostrato come questa possa essere utilizzata per risolvere sia il problema di Fault Detection che di Fault Isolation. L'obiettivo che ci poniamo adesso è quello di applicare questo strumento per l'individuazione e l'isolamento dei guasti sui thruster del veicolo V-Fides, descritto nel capitolo 2.

3.1 Acquisizione Dati

Come è già stato messo in evidenza nei paragrafi precedenti, la PCA è una tecnica model-free e per poterla applicare non è necessario, quindi, disporre di un modello del sistema, ma è sufficiente avere a disposizione un dataset costituito dalle variabili ritenute di maggior interesse per rappresentare il comportamento del sistema in condizioni operative.
Nel nostro caso, però, tale dataset non è disponibile e non possiamo perciò effettuare un'analisi delle componenti principali basata su dati sperimentali. Sorge, inoltre, anche il problema di cosa si intenda per condizioni operative nel caso di un veicolo subacqueo come V-Fides, il quale può esibire comportamenti molto diversi tra loro a seconda dell'operazione o della manovra che sta effettuando. È possibile, perciò, che un solo modello di PCA non sia sufficiente a catturare l'intera dinamica del sistema, ma che sia più opportuno anzi predisporre un insieme di modelli a seconda della legge di guida (e dei waypoint) al quale è sottoposto.
Infine non bisogna dimenticare che, prima di poter implementare l'algoritmo di FDI direttamente sul veicolo, è necessario testarne l'efficienza attraverso delle simulazioni effettuate al variare dello stato degli attuatori. Per tutte queste ragioni, è chiaro che, per applicare la PCA al sistema in esame, non possiamo fare a meno del modello descritto nel paragrafo 2.3.
In questo paragrafo ci occuperemo proprio di descrivere come il modello del veicolo (realizzato mediante Simulink) possa essere utilizzato per elaborare e testare l'algoritmo di FDI tramite PCA su V-Fides.

Il primo problema che ci troviamo ad affrontare è la scelta delle condizioni operative sulla base delle quali determinare le componenti principali del sistema. Sappiamo che il sistema presenta 3 gradi di libertà, dal momento che, come descritto nel paragrafo 2.2, le velocità di *roll, pitch* e *sway* sono forzate a zero dall'autopilota. Ciò che può cambiare in modo significativo da una missione all'altra, invece, sono le velocità desiderate di *surge, heave* e *yaw*, che vengono calcolate dal sistema di guida. La nostra scelta è stata quella di studiare il veicolo in presenza di una legge di guida in grado di condurlo lungo un esempio tipico di survey. Per semplicità si è supposto che durante la survey il veicolo non modifichi mai la sua profondità (velocità di $heave = 0\ [m/s]$), in modo che il suo moto avvenga completamente lungo il piano $X-Y$. In figura 3.1 sono evidenziati gli "waypoint" (rosso) che definiscono la survey scelta.

Il sistema di guida non si occuperà solo di calcolare le velocità di riferimento tali da direzionare il veicolo verso il waypoint successivo, ma cercherà anche di far sì che la traiettoria venga completata nel minor tempo possibile. Per questo motivo la velocità desiderata di avanzamento sarà proprio pari a $1\ [m/s]$, la massima consentita per V-Fides.
È evidente che la traiettoria in figura 3.1 non potrà essere realisticamente seguita in modo perfetto

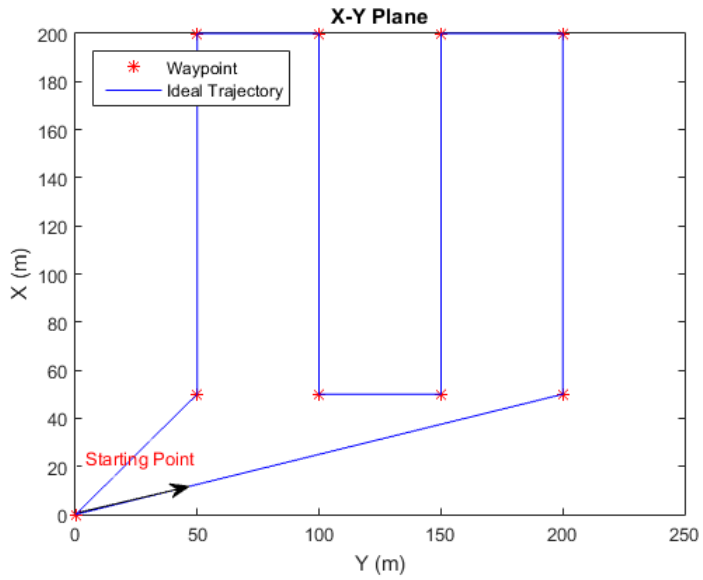


Figura 3.1: Waypoint (rosso) e traiettoria desiderata lungo il piano $X-Y$ durante la survey (blu).

dal momento che non tiene in considerazione il minimo raggio di curvatura possibile per il veicolo. Verosimilmente il percorso seguito assomiglierà maggiormente a quello mostrato in figura 3.2, il quale è stato ottenuto simulando il comportamento del veicolo in condizioni ideali, ovvero assenza di disturbi e guasti sugli attuatori.

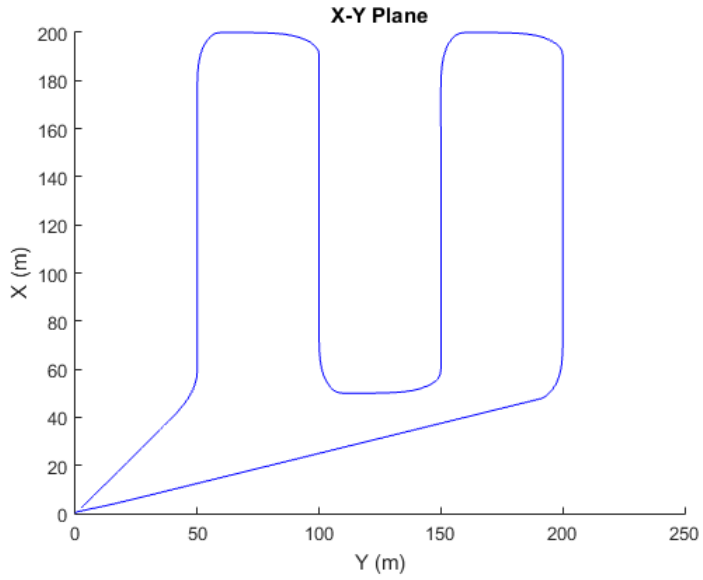


Figura 3.2: Traiettoria lungo il piano $X-Y$ percorsa dal veicolo durante la survey, in assenza di guasti sugli attuatori.

Nelle figure 3.3 e 3.4 sono riportati gli andamenti delle velocità ideali rispettivamente di *surge* e *yaw* imposte dal sistema di guida affinché il veicolo possa seguire la survey descritta.

Il sistema è stato simulato per 1200 $[s]$ con un tempo di campionamento $T_s = 0.1$ $[s]$.
Dal grafico in figura 3.3, si può osservare che la velocità di avanzamento aumenta rapidamente fino a raggiungere il valore di 1 $[m/s]$ e si mantiene costante per quasi 1000 $[s]$, trascorsi i quali il veicolo è riuscito a terminare la sua traiettoria e può quindi rallentare fino a fermarsi in corrispondenza

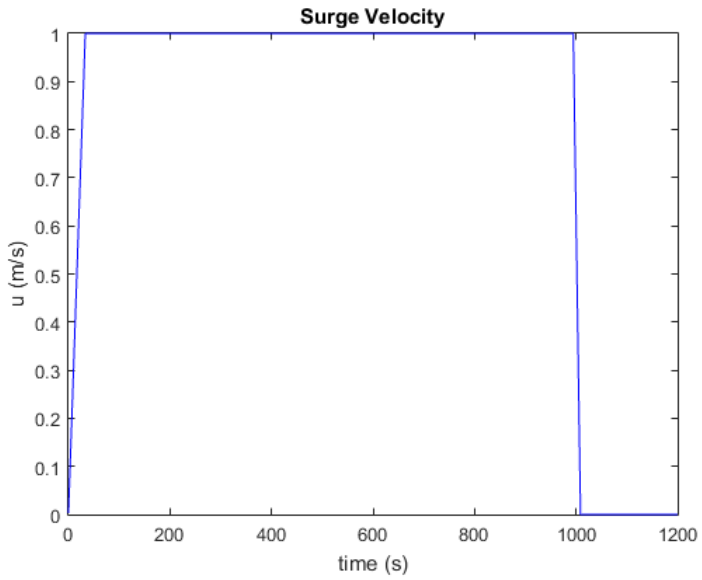


Figura 3.3: Velocità di avanzamento desiderata.

dell'ultimo waypoint. Per rendere lo scenario più realistico, supponiamo che il veicolo inizi ad acquisire dati solo dopo aver raggiunto pieno regime (circa 100 $[s]$ dall'inizio della survey) e che interrompa l'acquisizione una volta ritornato alla base (dopo circa 1000 $[s]$ di simulazione). Questo è giustificato dal fatto che la probabilità che avvenga un guasto sul sistema non è trascurabile solo quando il veicolo si sta muovendo autonomamente lungo il suo cammino, mentre è altamente improbabile che un thruster si rompa quando questo è fermo in un punto noto.
Dal momento che il percorso è per la maggior parte rettilineo, la velocità angolare di imbardata è sempre nulla, tranne in qualche breve intervallo in cui cambia per consentire al veicolo di curvare.

Una volta definite le condizioni nominali nelle quali intendiamo far operare il veicolo, è ne-

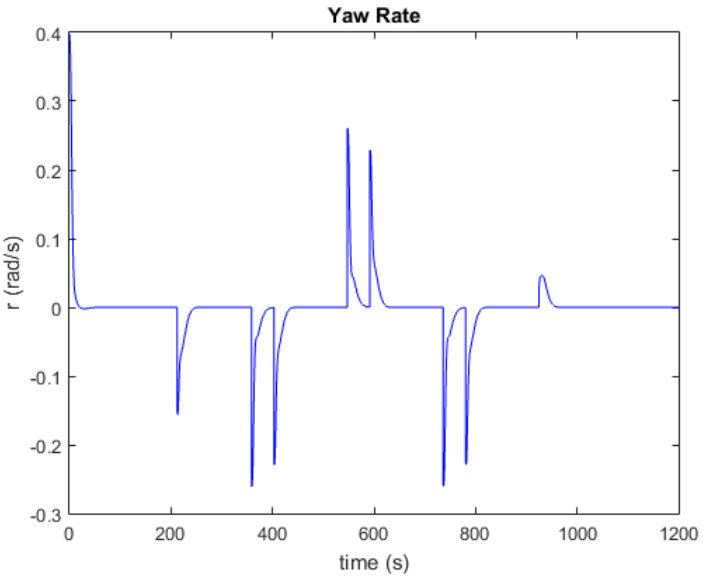


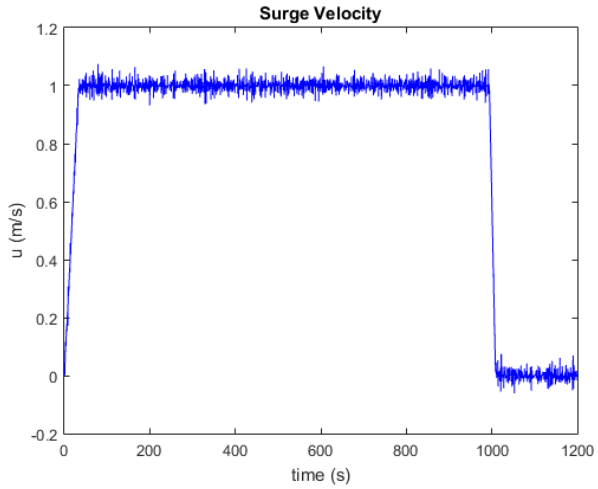
Figura 3.4: Velocità di imbardata desiderata.

cessario stabilire quali variabili utilizzare per costruire il dataset che ci permetterà di effettuare un'analisi delle componenti principali del sistema.
Nel sistema reale non tutte le variabili possono essere misurate o stimate in modo accurato, per questo motivo alcune scelte che potrebbero sembrare più logiche dal punto di vista teorico, non possono essere prese in considerazione per ragioni pratiche. In prima battuta, però, possiamo decidere di trascurare questo aspetto, dal momento che, fintantoché siamo in fase di simulazione, tutte le variabili possono essere calcolate e memorizzate. I dati ritenuti maggiormente significativi per i nostri scopi sono:

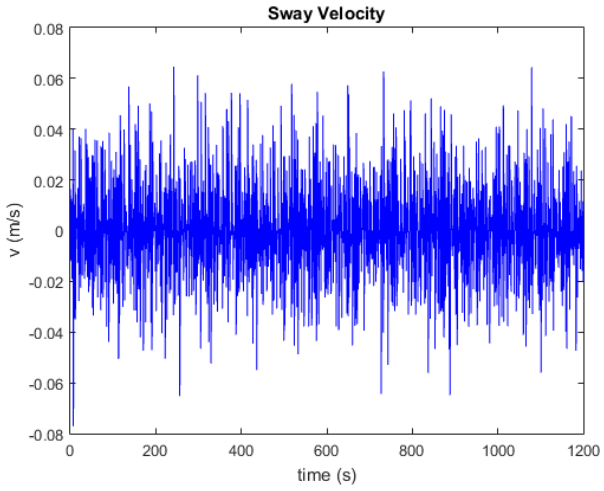
- $f \in \mathbb{R}^7 \rightarrow$ Vettore contenente le forze esercitate dai thruster;
- $u_{cmd} \in \mathbb{R}^7 \rightarrow$ Vettore contenente i comandi in ingresso agli attuatori;
- $\nu \in \mathbb{R}^6 \rightarrow$ Velocità lineari e angolari in *body frame*;
- $\eta \in \mathbb{R}^6 \rightarrow$ Posizioni e angoli di Eulero in *navigation frame*.

Memorizzando ad ogni istante di campionamento della finestra temporale selezionata (tra 100 $[s]$ e 1000 $[s]$) tutte le variabili di interesse, otteniamo un dataset costituito da 9000 misure di 26 variabili. La scelta di quante e quali variabili utilizzare durante la fase di elaborazione *off-line* sarà discussa nel paragrafo successivo.
È bene sottolineare, però, che, data la natura non omogenea delle variabili all'interno del dataset, queste devono essere sempre normalizzate; la PCA richiede, infatti, dati a media nulla e varianza unitaria sia durante l'elaborazione *off-line* sia durante il monitoraggio *on-line*.
Proprio per questo motivo, per applicare correttamente l'algoritmo è fondamentale disporre di una stima accurata delle caratteristiche del rumore agente sul sistema. Questo, infatti, potrebbe introdurre dei bias o delle variazioni significative sulla deviazioni standard delle variabili utilizzate, che, se non tenute adeguatamente in considerazione, potrebbero portare ad un'errata normalizzazione dei dati.
Al fine di rendere lo scenario descritto più realistico, abbiamo quindi deciso di corrompere le variabili acquisite con l'aggiunta di rumore gaussiano bianco. In questo modo si sono potuti prendere in considerazione i possibili effetti dovuti alla presenza del rumore sui sensori installati a bordo del veicolo. In particolare si è scelto di utilizzare sia per le velocità lineari che angolari del rumore gaussiano bianco a media nulla e con deviazioni standard rispettivamente di 0.01 e 0.005 . Tali valori, determinati sulla base di dati sperimentali acquisiti su altri veicoli subacquei, rappresentano, in realtà, una stima per eccesso rispetto a quelli effettivamente osservabili su V-Fides, dal momento che quest'ultimo è dotato di unità inerziali estremamente accurate.
In figura 3.5 sono mostrati i dati rumorosi acquisiti durante la simulazione.

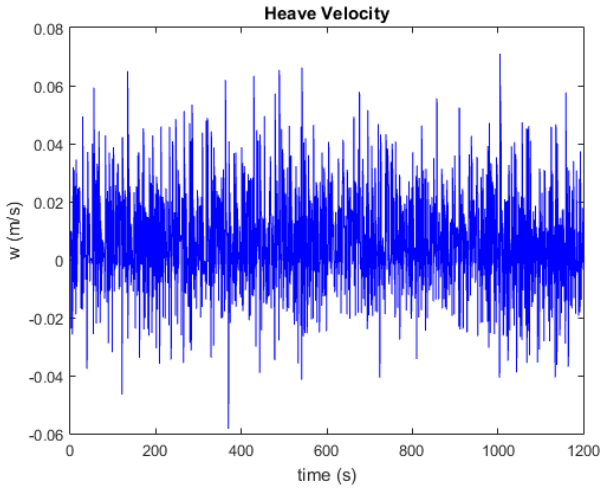
Ricordiamo che ogni modello PCA è fortemente dipendente dai dati con i quali è stato elaborato, per questo motivo non vi è alcuna garanzia che l' algoritmo sia in grado di risolvere il problema di FDI su una traiettoria diversa da quella con cui è stato addestrato. Nel paragrafo 5.3 verrà mostrato come in realtà, l'algoritmo disponga di una buona capacità di generalizzazione, risultando applicabile anche in survey differenti da quella proposta in questo paragrafo. È chiaro, però, che manovre come il cambiamento di profondità (moto non confinato al piano $X-Y$) o curve molto strette (rotazioni dell'angolo di imbardata maggiori di 90°), porteranno probabilmente alla generazione di falsi allarmi, dal momento che nessuna di queste operazioni è stata inclusa nel dataset con il quale l'algoritmo è stato addestrato.
Abbiamo quindi due possibili soluzioni per risolvere questo problema ovvero cercare di includere tali manovre nella survey utilizzata per addestrare il veicolo, oppure predisporre più modelli PCA del sistema a seconda del tipo di missione che il veicolo sta effettuando.



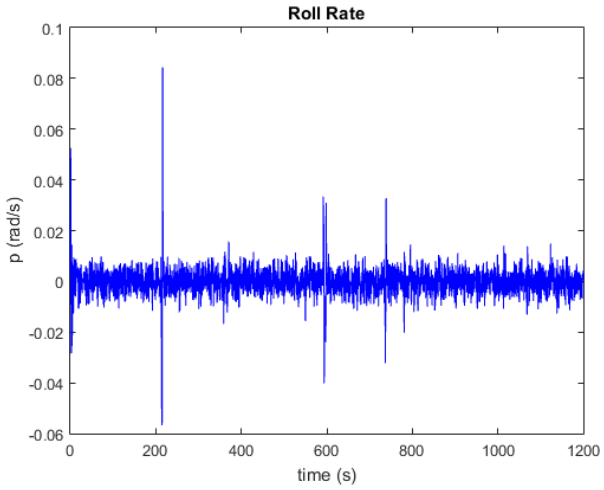
(a) *Velocità di avanzamento (u).*



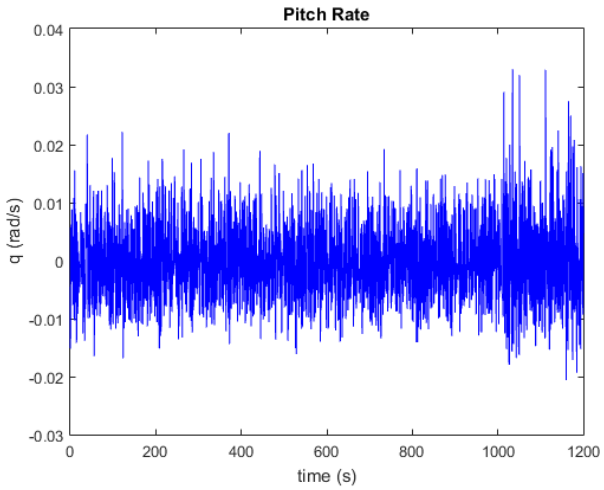
(b) *Velocità laterale (v).*



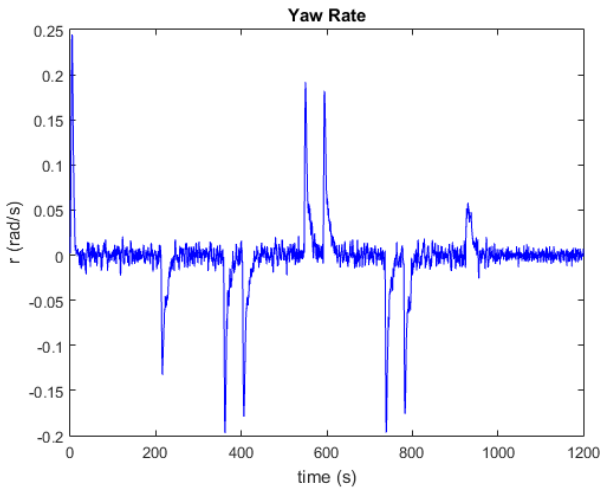
(c) *Velocità verticale (w).*



(d) *Velocità di rollio (p).*



(e) *Velocità di beccheggio (q).*



(f) *Velocità di imbardata (r).*

Figura 3.5: Velocità del veicolo durante la simulazione con l'aggiunta di rumore gaussiano bianco.

3.2 Guasti sui thruster

In questo paragrafo verrà presentata brevemente l'influenza dei vari thruster nel portare a termine la survey descritta nel paragrafo precedente e verranno messi in risalto gli effetti sul sistema di un loro possibile guasto.
La collocazione di ogni thruster è mostrata nell'immagine in figura 2.1.

3.2.1 Thruster 1 e 2

I thruster 1 e 2, disposti a poppa, sono gli unici in grado di generare una forza positiva lungo la direzione di *surge* e per questo motivo giocano un ruolo fondamentale per l'avanzamento del veicolo. Nelle figure 3.6 e 3.7 sono mostrati gli andamenti delle forze generate dai due thruster durante la survey.

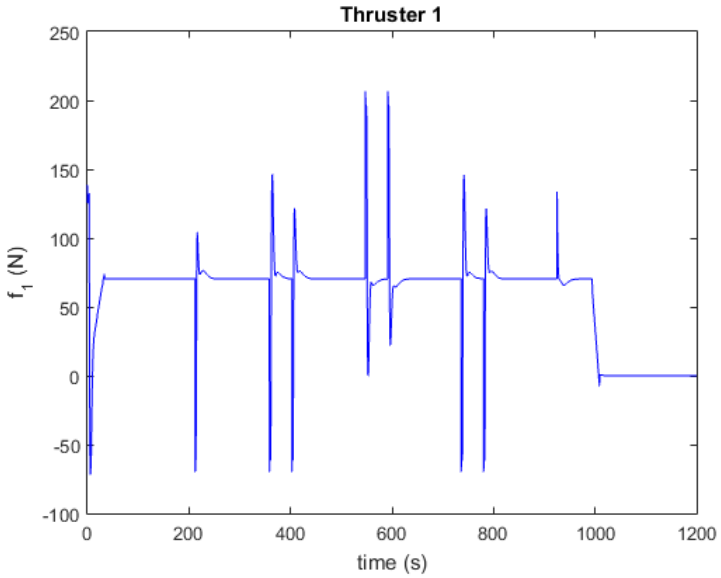


Figura 3.6: Forza generata dal thruster 1 durante la survey (in assenza di rumore e guasti).

Possiamo osservare che il comportamento dei due attuatori è molto simile. Entrambi, infatti, generano delle forze di circa 70 $[N]$ durante i tratti rettilinei della traiettoria, mentre presentano dei "picchi" opposti e speculari, in corrispondenza delle "curve". Questo è dovuto al fatto che, data la loro posizione simmetrica rispetto all'asse longitudinale di V-Fides, quando i due attuatori spingono nella stessa direzione, producono una coppia di imbardata uguale e opposta l'uno rispetto all'altro.
Una volta che la velocità di avanzamento del veicolo scende a zero, l'azione dei due attuatori si annulla. Per questo motivo se un eventuale guasto sui due thruster avvenisse dopo 1000 $[s]$ dall'inizio della survey, difficilmente potrebbe essere rilevato e sicuramente non andrebbe ad alterare la stabilità del veicolo.
Molto diversa sarebbe la situazione se, invece, il guasto avvenisse prima. In questo caso, infatti, senza un opportuno sistema di *fault recovery*, il sistema non sarebbe in grado di completare la survey. In figura 3.8 è mostrato cosa succede alle velocità di *surge* e *yaw* in presenza di un guasto

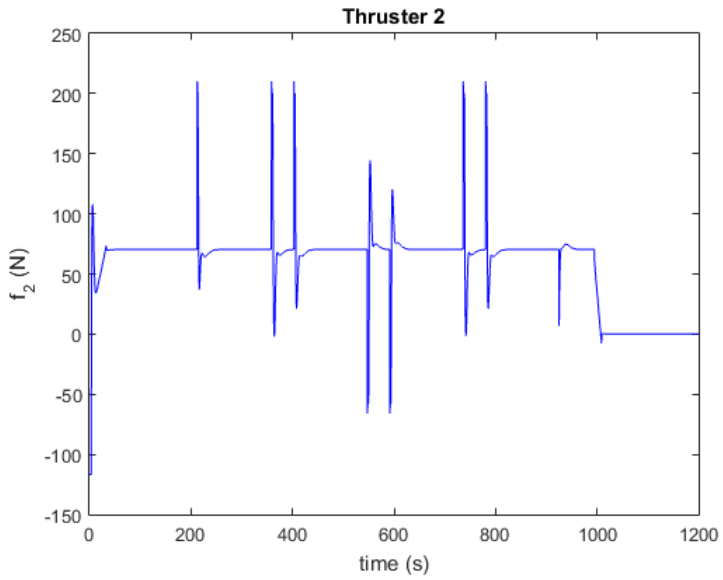
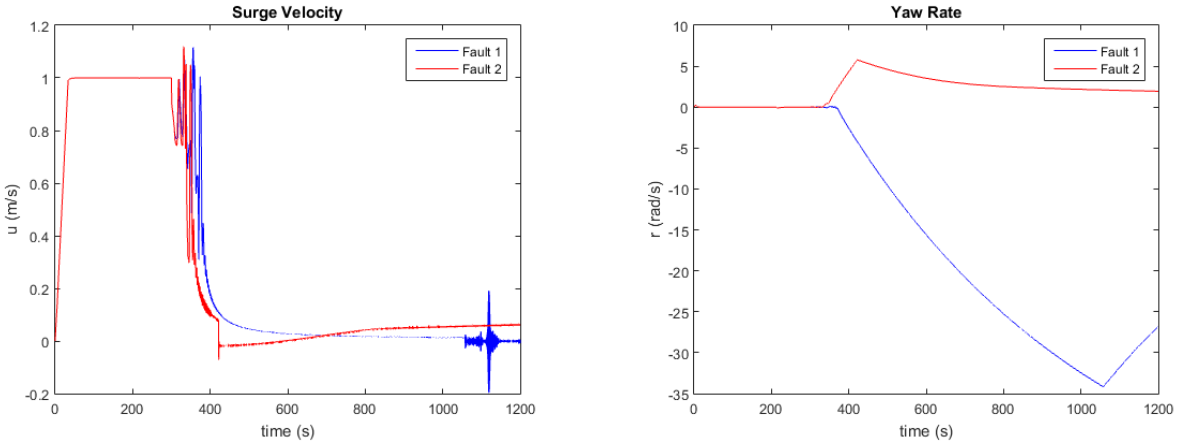


Figura 3.7: Forza generata dal thruster 2 durante la survey (in assenza di rumore e guasti).

su uno dei due attuatori (ogni guasto è stato simulato dopo 300 $[s]$ dall'inizio della survey).



(a) *Velocità di avanzamento (u).* (b) *Velocità di imbardata (r).*

Figura 3.8: Velocità di avanzamento e imbardata in presenza di un guasto sul primo (blu) e sul secondo attuatore (rosso). in entrambi i casi il guasto è stato simulato a partire dall'istante $t = 300\ s$.

In entrambi i casi la velocità di *surge* scende rapidamente a zero, mentre il veicolo inizia a ruotare su sé stesso (in direzione diversa a seconda del guasto).

3.2.2 Thruster 3

Il thruster 3, disposto a prua del veicolo e con asse di azione parallela a quella di *sway*, agisce in prevalenza generando una forza lungo l'asse laterale e una coppia di *yaw*. Come possiamo notare dal grafico in figura 3.9, questo è probabilmente l'attuatore meno utilizzato durante la survey.

La forza generata dal thruster, infatti, risulta nulla per quasi tutta la traiettoria, raggiungendo valori diversi da zero solo quando al veicolo è richiesto di virare. In figura 3.10, sono messe a

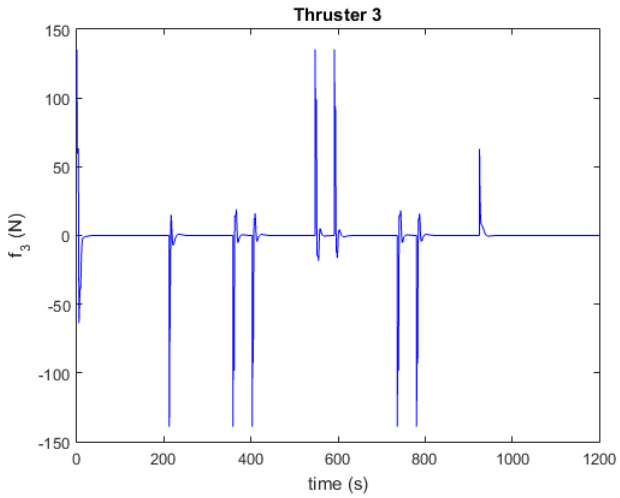
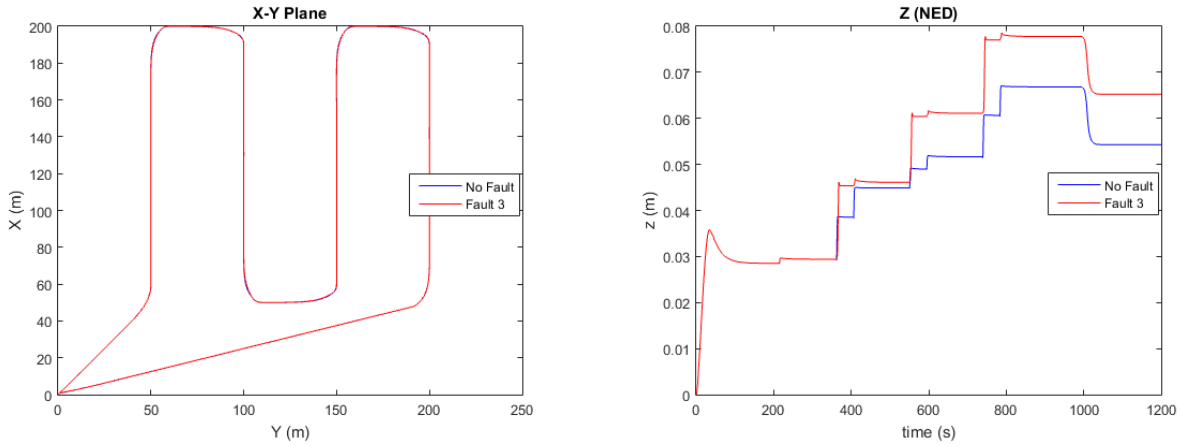


Figura 3.9: Forza generata dal thruster 3 durante la survey (in assenza di rumore e guasti).

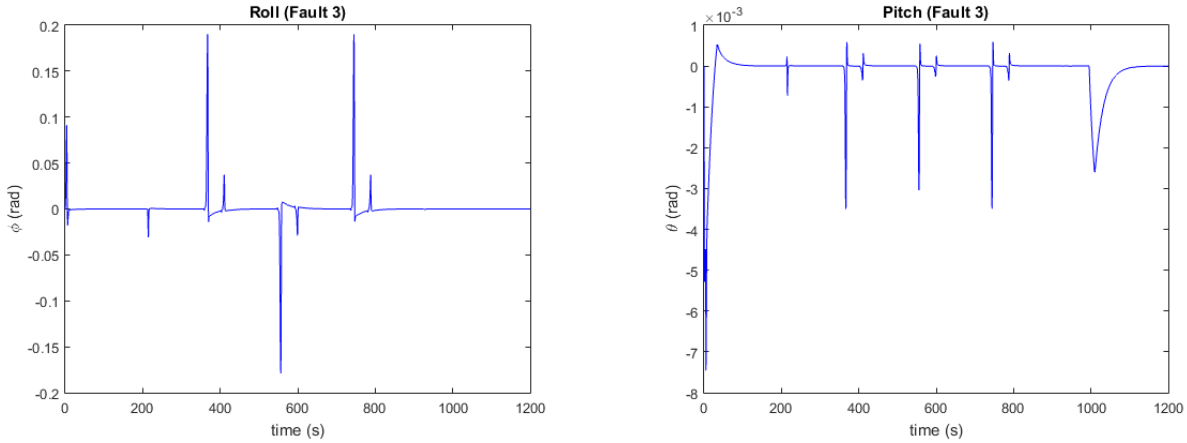
confronto i percorsi effettuati dal veicolo durante la survey sia in assenza di guasti, che in presenza di un guasto sul III attuatore (simulato a partire dall'istante $t = 300\ [s]$).



(a) *Traiettoria sul piano $X-Y$.* (b) *profondità al variare del tempo.*

Figura 3.10: Traiettoria sul piano $X-Y$ e profondità del veicolo durante la survey (in assenza di rumore) sia in assenza di guasti (blu), sia in presenza di un guasto sul thruster 3 (rosso) simulato a partire dall'istante $t = 300\ s$.

È possibile notare che le due traiettorie sul piano $X-Y$ risultano quasi sovrapposte, mentre per la profondità si ha un errore dell'ordine dei $[cm]$ e tutto questo senza che sia stato implementato alcun sistema di fault recovery. Inoltre il veicolo riesce anche a mantenere un assetto stabile anche quando l'attuatore 3 si rompe (figura 3.11).
La bassa influenza di un guasto di questo tipo sulle prestazioni del sistema è da una parte positiva, dal momento che non compromette la riuscita della missione, dall'altra, però, potrebbe rendere difficile ottenere informazioni sullo stato del thruster e portare ad un mancato riconoscimento di una sua eventuale rottura.



(a) *Roll* (ϕ). (b) *Pitch* (θ).

Figura 3.11: Assetto del veicolo durante la survey (in assenza di rumore) in presenza di un guasto sul thruster 3 simulato a partire dall'istante $t = 300\ s$.

3.2.3 Thruster 4 e 5

I thruster 4 e 5 sono entrambi disposti a poppa e con asse in direzione di *heave*. Nelle figure 3.12 e 3.13 sono mostrati gli andamenti delle forze generate dai due thruster durante la survey.

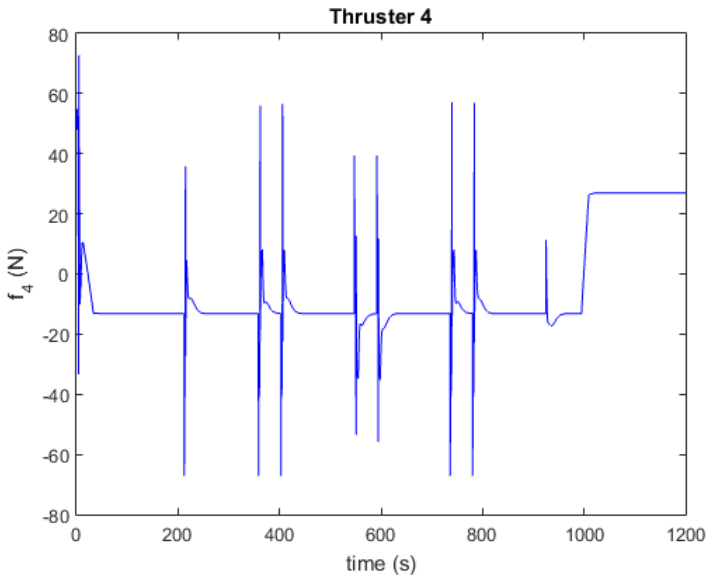


Figura 3.12: Forza generata dal thruster 4 durante la survey (in assenza di rumore e guasti).

I due attuatori giocano un ruolo fondamentale nel mantenere sotto controllo l'assetto del veicolo, per questo motivo vengono sempre stimolati anche quando la survey è terminata e il veicolo si ferma. Senza un modulo opportuno di fault recovery, il sistema, se soggetto ad uno di questi due guasti, diventa rapidamente instabile (figure 3.14 e 3.15).

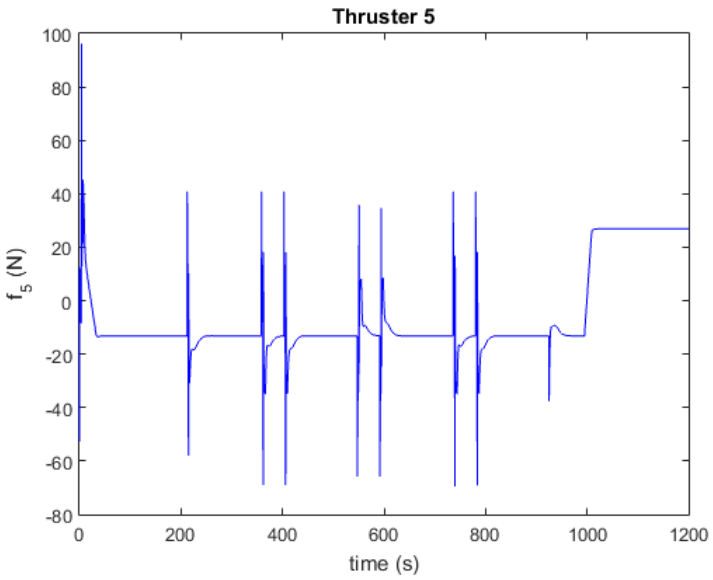
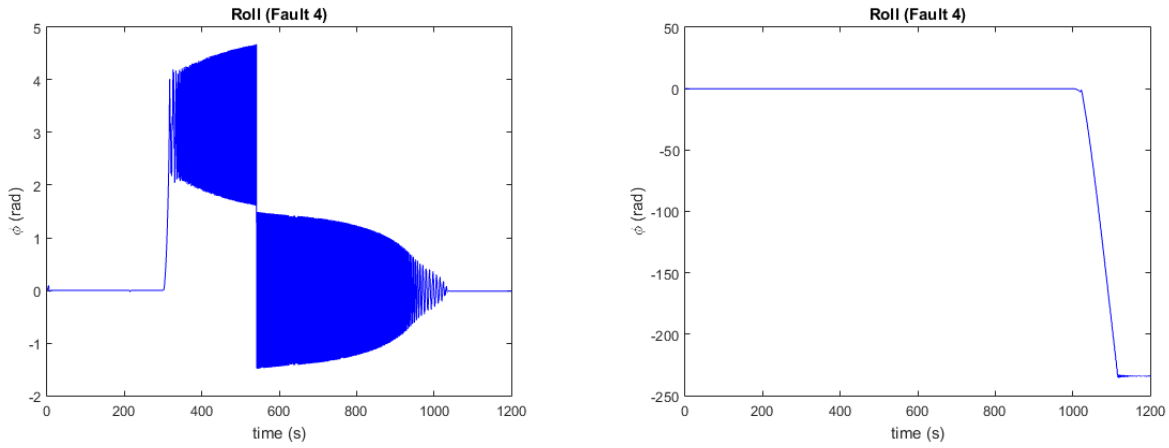


Figura 3.13: Forza generata dal thruster 5 durante la survey (in assenza di rumore e guasti).

3.2.4 Thruster 6

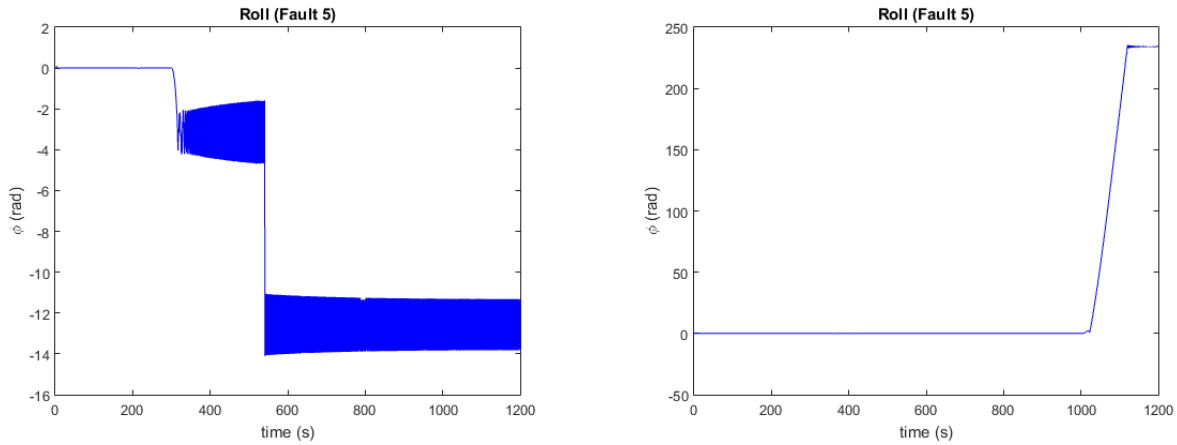
Il thruster 6 è collocato a prua e come il 4 e il 5, ha l'asse di azione verticale. In figura 3.16 è riportato l'andamento nel tempo della forza generata dall'attuatore durante la survey e in assenza di guasti.

Questo attuatore è il più critico tra tutti quelli presenti su V-Fides. Quando il veicolo si sta muovendo a velocità contenute, gli effetti destabilizzanti delle forze idrodinamiche agenti sulla parte frontale del veicolo tendono a generare delle coppie di beccheggio che sono controbilanciate soltanto dall'azione di questo attuatore, in quanto la coppia di thrusters 4-5 (disposta a poppa) non è sufficiente a garantire il controllo di assetto. Pertanto, una rottura di questo attuatore non



(a) *Guasto a partire dall'istante* $t = 300\ s$. (b) *Guasto a partire dall'istante* $t = 1000\ s$.

Figura 3.14: Angolo di rollio ϕ durante la survey, in presenza di un guasto sul quarto attuatore e in assenza di rumore.



(a) *Guasto a partire dall'istante* $t = 300$ *s*. (b) *Guasto a partire dall'istante* $t = 1000$ *s*.

Figura 3.15: Angolo di rollio ϕ durante la survey, in presenza di un guasto sul quinto attuatore e in assenza di rumore.

può essere in alcun modo compensata. Questo, in realtà, non succede quando il veicolo si sta muovendo a velocità molto elevate (es: 1 $[m/s]$). In tali condizioni, infatti, l'azione degli altri thruster riesce a mantenere il veicolo stabile, come possiamo osservare in figura 3.17, nella quale è riportato l'andamento nel tempo dell'angolo di beccheggio, in presenza di un guasto simulato a partire dall'istante $t = 300$ $[s]$. Possiamo osservare come il sistema risenta della rottura del thruster, raggiungendo angoli di beccheggio "pericolosi", solo dopo i 1000 $[s]$, ovvero quando il veicolo inizia a rallentare.

Se si muove ad alte velocità, il veicolo è quindi in grado di portare a termine la missione anche in assenza del sesto attuatore, a discapito però di un significativo aumento di profondità (figura 3.18).

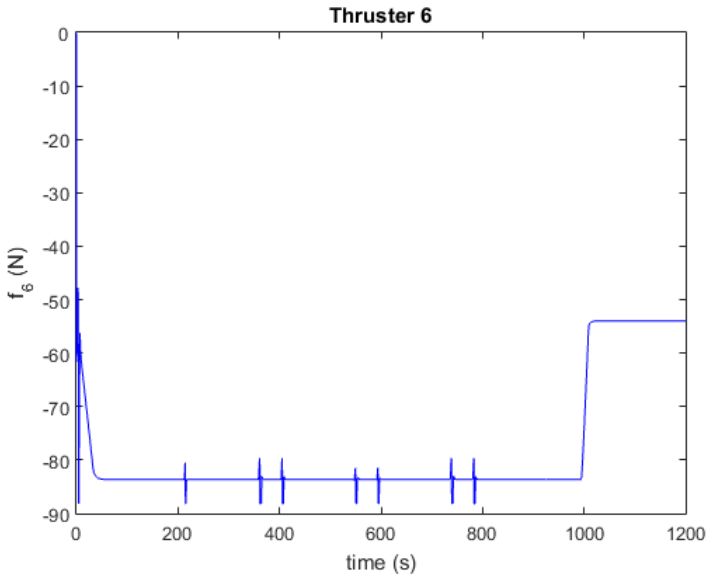


Figura 3.16: Forza generata dal thruster 6 durante la survey (in assenza di rumore e guasti).

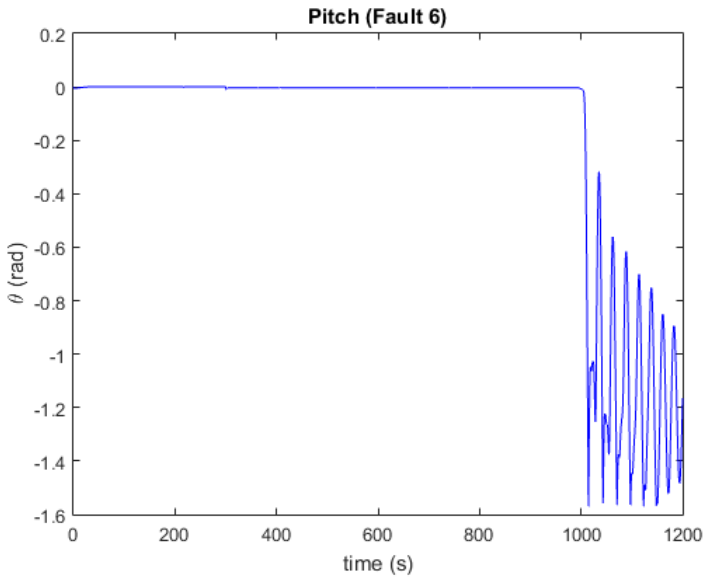
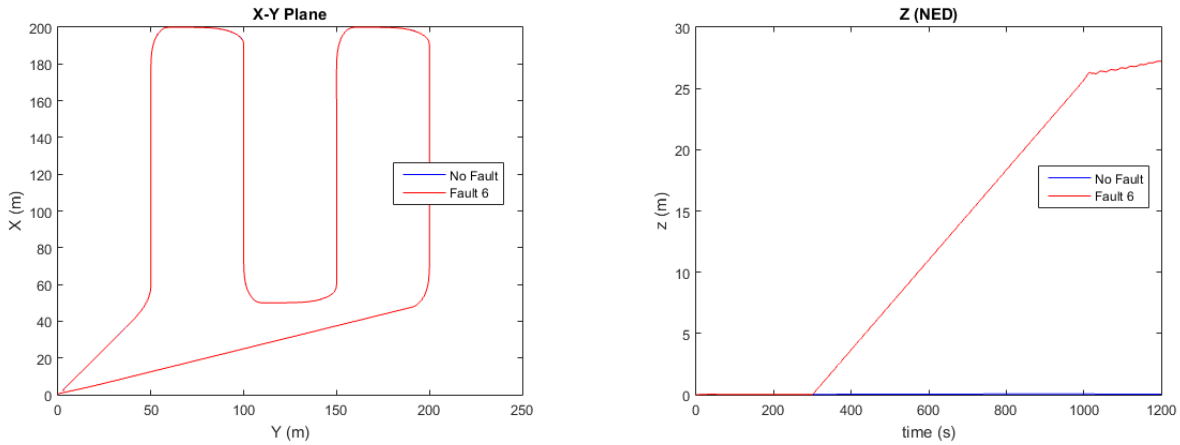


Figura 3.17: Angolo di beccheggio del veicolo durante la survey (in assenza di rumore) in presenza di un guasto sul sesto attuatore simulato a partire dall'istante $t = 300\ s$.

3.2.5 Thruster 7

L'attuatore 7 è posizionato sulla "pinna" ed è in grado di controllare la velocità di *sway* e fornire coppie di rollio e imbardata. Come il thruster 3, questo è quello che influisce meno sulla dinamica del sistema, mantenendosi a zero per quasi tutta la durata della survey (figura 3.19).

A differenza di quanto osservato per i guasti sul terzo attuatore, però, un guasto sul settimo impedisce al veicolo di portare a termine la survey. Infatti, come possiamo osservare dal grafico in figura 3.20, senza un opportuno sistema di fault recovery, il veicolo si destabilizza e finisce per ribaltarsi.



(a) *Traiettoria sul piano $X - Y$.* (b) *profondità al variare del tempo.*

Figura 3.18: Traiettoria sul piano $X - Y$ e profondità del veicolo durante la survey sia in assenza di guasti (blu), che in presenza di un guasto sul thruster 6 (rosso) simulato a partire dall'istante $t = 300\ s$.

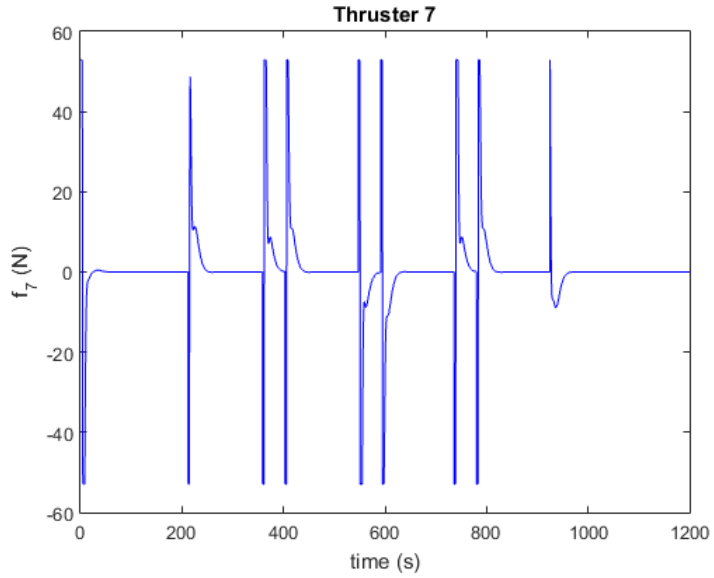


Figura 3.19: Forza generata dal thruster 7 durante la survey (in assenza di rumore e guasti).

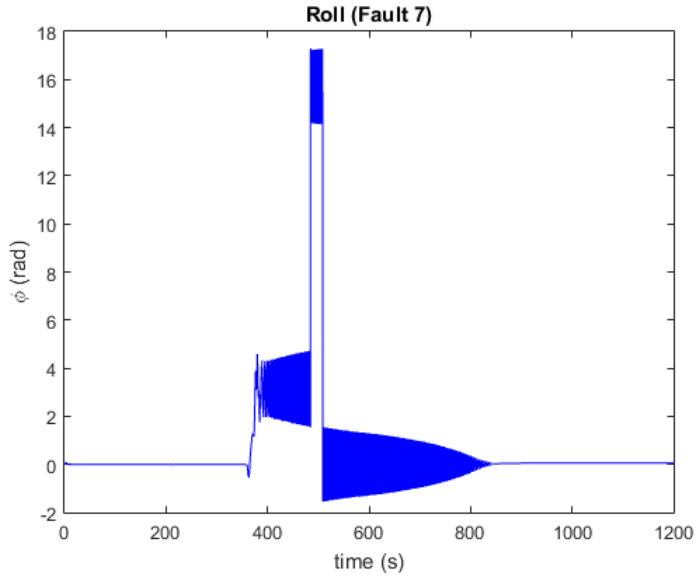


Figura 3.20: Angolo di rollio ϕ durante la survey, in presenza di un guasto sul thruster 7 simulato a partire dall'istante $t = 300\ s$.

3.3 PCA Standard

In base a quanto osservato nel paragrafo 1.3, per poter applicare la versione standard dell'algoritmo di FDI con PCA, è necessario che vi sia una corrispondenza 1 : 1 tra variabili del sistema e guasti. Purtroppo, dal momento che non è possibile misurare le forze esercitate dagli attuatori, tale corrispondenza non può essere ottenuta. Un'idea possibile per poter ovviare a questo problema, è quella di fornire delle direzioni di ricostruzione "alternative" rispetto a quelle considerate finora. Supponiamo, infatti, di avere come vettore dei dati il vettore $\nu \in \mathbb{R}^6$, delle velocità lineari e angolari espresse in *body frame*. Queste devono essere poste in relazione con il vettore f delle forze esercitate dai thruster, che però "vive" in uno spazio a dimensione 7. Dall'equazione 2.9, possiamo osservare che la matrice $B \in \mathbb{R}^{6\times7}$, effettua una mappatura proprio tra lo "spazio delle forze" al quale appartiene f e il *body frame* nel quale si trova ν. È possibile supporre, quindi, che un guasto sul thruster i, andrà a influenzare le variabili del sistema nel modo previsto dall'i-esima colonna b_i della matrice B. Le colonne normalizzate di B, saranno, perciò, le direzioni di ricostruzione dei fault considerati.
Prima di testare la validità di questo ragionamento, può essere però interessante capire cosa succederebbe se avessimo a disposizione direttamente il vettore f. Infatti tale misura è accessibile fintanto che siamo in fase di simulazione.

Consideriamo un dataset $X \in \mathbb{R}^{9000\times7}$, ottenuto secondo la procedura descritta nel paragrafo 3.1 e considerando le forze f come uniche variabili di interesse.
In accordo con l'algoritmo presentato nel paragrafo 1.6, procediamo con l'elaborazione *off-line* dei dati ed in particolare con la scelta del numero l delle componenti principali.
Osservando la curva in figura 3.21, che riporta l'andamento dell'indice RPV (equ. 1.21), possiamo notare la presenza di un "ginocchio" in corrispondenza di $l = 2$. Tale punto rappresenta proprio la scelta suggerita dal criterio e permette di catturare oltre il 95% della varianza dei dati (come mostrato in tabella 3.1).

A rafforzare questa tesi, troviamo anche il criterio della *Parallel Analysis* (figura 3.22). Solo i primi due autovalori della matrice di correlazione dei dati, infatti, risultano maggiori di quelli ottenuti da un dataset di variabili tra loro non correlate.

Passiamo, adesso, alla modellazione dei guasti. Per semplicità ci andremo ad occupare solo di fault singoli sugli attuatori. Questa ipotesi non risulta particolarmente stringente dal momento che la rottura di più thruster può essere considerato un evento piuttosto raro e con scarse possibilità di

l	$CPV(\%)$
1	63.4
2	97.4
3	98.9
4	99.6
5	99.8
6	99.9
7	100

Tabella 3.1: Cumulative Percent Variance con vettore delle forze.

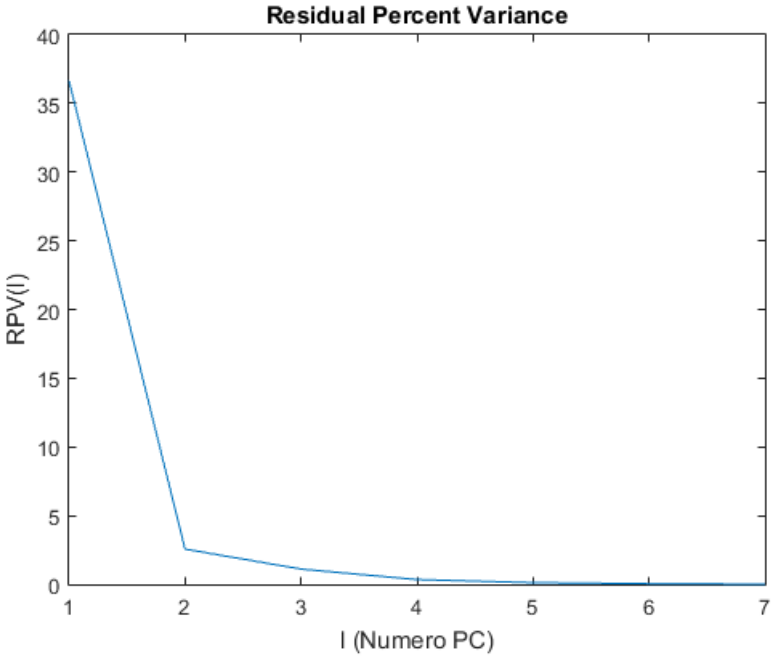


Figura 3.21: Residual Percent Variance.

$\tilde{\Xi}_1^T\tilde{\Xi}_1$	$\tilde{\Xi}_2^T\tilde{\Xi}_2$	$\tilde{\Xi}_3^T\tilde{\Xi}_3$	$\tilde{\Xi}_4^T\tilde{\Xi}_4$	$\tilde{\Xi}_5^T\tilde{\Xi}_5$	$\tilde{\Xi}_6^T\tilde{\Xi}_6$	$\tilde{\Xi}_7^T\tilde{\Xi}_7$
0.286	0.281	0.870	0.871	0.868	0.938	0.887

Tabella 3.2: Studio rilevabilità dei guasti.

recupero. Se due motori si dovessero rompere, infatti, il sistema di controllo difficilmente potrebbe compensare tale perdita, anche nel caso in cui questi venissero identificati correttamente. Nel caso di guasto singolo, invece, un modulo di FDI assumerebbe un ruolo fondamentale, perché il sistema di controllo potrebbe modificare l'azione degli altri attuatori permettendo al veicolo di portare a termine la missione (seppur con prestazioni ridotte). Questo è possibile poiché V-Fides è sovra-attuato e quindi, anche in caso di rottura di un motore, avrebbe a disposizione altri 6 thruster su cui fare affidamento.
I guasti possibili sono perciò 7 e il generico guasto i è modellato con una matrice $\Xi_i \in \mathbb{R}^7$ costituita da tutti zeri e un 1 in posizione i. Applicando l'algoritmo per lo studio della rilevabilità e isolabilità dei guasti (paragrafo 1.4), otteniamo i risultati mostrati nelle tabelle 3.2 e 3.3.

Rcond	1	2	3	4	5	6	7
1	×	0.967	0.478	0.539	0.423	0.512	0.482
2	×	×	0.470	0.419	0.533	0.506	0.485
3	×	×	×	0.949	0.950	0.963	0.870
4	×	×	×	×	0.892	0.904	0.956
5	×	×	×	×	×	0.903	0.949
6	×	×	×	×	×	×	0.973

Tabella 3.3: Studio isolabilità dei guasti.

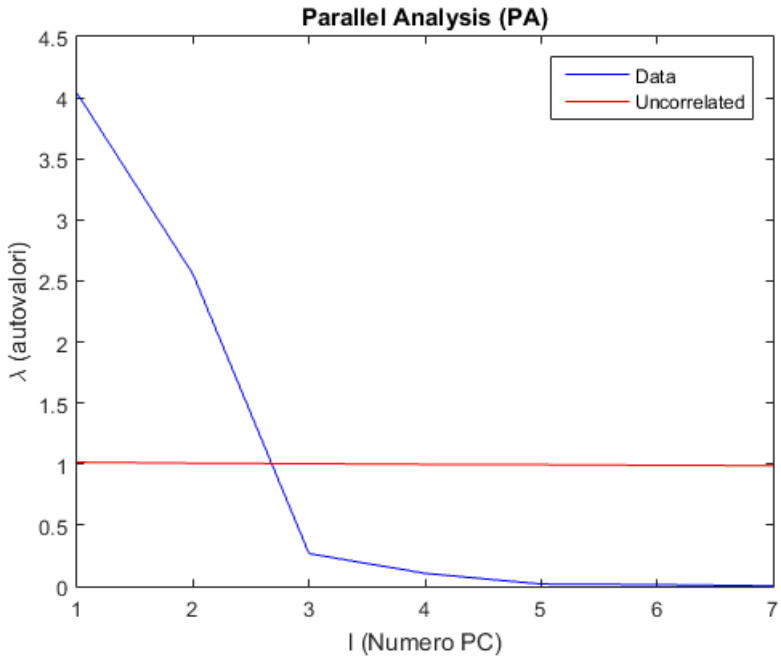
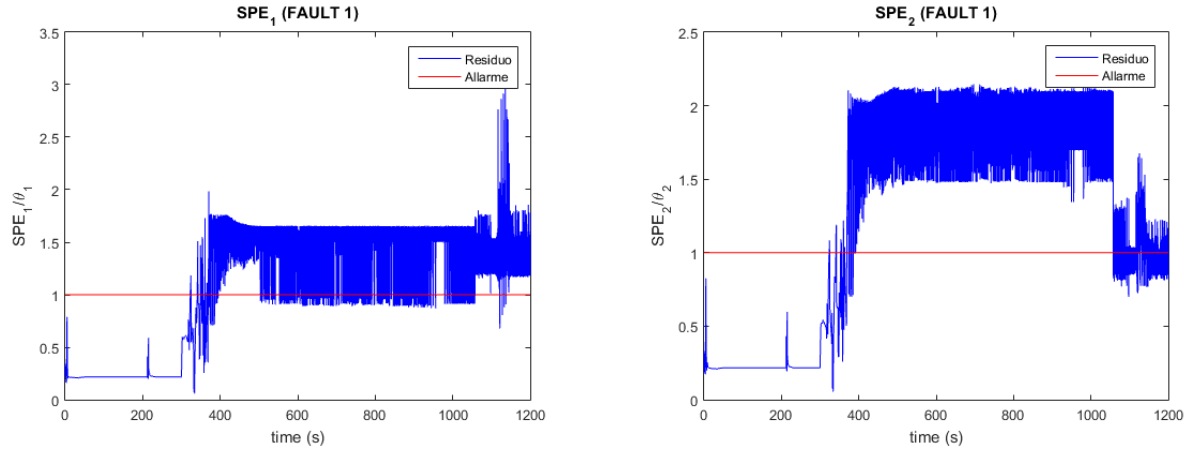


Figura 3.22: Confronto tra gli autovalori della matrice di correlazione dei dati (blu) e quelli otttenuti da un dataset costituito da variabili tra loro non correlate (rosso).

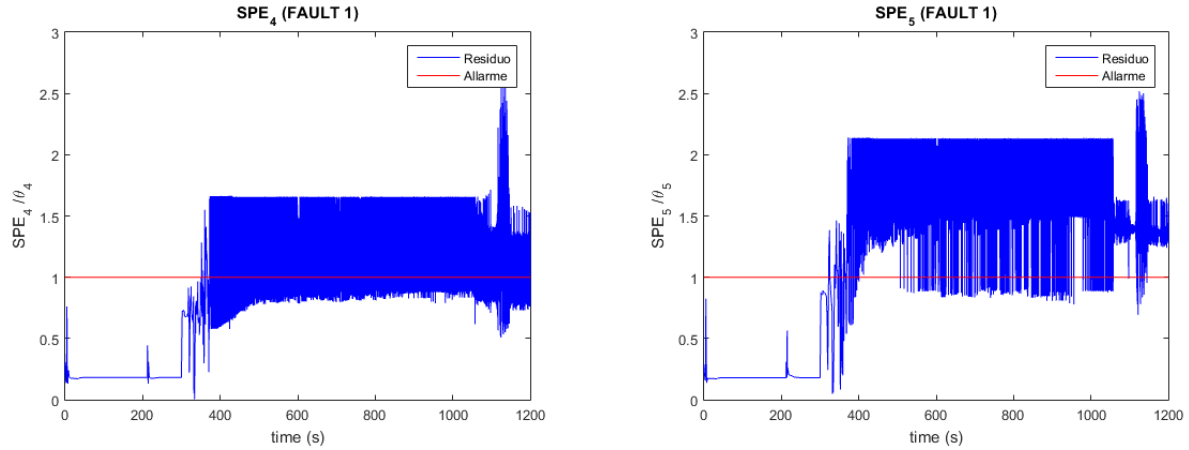
Dal momento che nessuno dei valori in tabella risulta particolarmente piccolo rispetto agli altri, possiamo concludere che tutti i guasti sono rilevabili e isolabili l'uno rispetto all'altro.
Prima di procedere con la verifica *on-line* dell'efficacia dell'algoritmo di FDI, è stata fissata, per ogni residuo SPE_i, una soglia θ_i basata sui valori assunti dallo stesso in condizioni nominali.

Nelle figure 3.23 e 3.24, sono rappresentati gli andamenti nel tempo di alcuni dei residui, nel caso in cui sia stato simulato rispettivamente un guasto sul primo e sul quarto attuatore. In entrambi i casi, il guasto è avvenuto dopo 300 s.

Possiamo osservare che, in entrambi i casi, i guasti sono stati rilevati correttamente, infatti il segnale in rosso (allarme) viene superato dal residuo dopo circa 50 s dall'inserimento del guasto. Se osserviamo bene i grafici, però, ci accorgiamo che, differentemente da quanto atteso, tutti i residui superano la soglia d'allarme. Secondo l'algoritmo per la Fault Isolation tramite PCA, il residuo associato al guasto realmente avvenuto dovrebbe mantenersi sotto la soglia, invece nel nostro caso non succede. Infatti sia SPE_1 in figura 3.23 che SPE_4 in figura 3.24, dopo un breve transitorio, raggiungono valori al di sopra della soglia critica. Stesso comportamento è stato riscontrato per tutti i guasti e per tutti i residui (anche quelli il cui andamento non è stato riportato).
L'algoritmo così proposto, perciò, pur rilevando correttamente la presenza del guasto, non riesce in alcun modo ad isolarlo. La spiegazione di questo fenomeno può essere compresa osservando i grafici in figura 3.25. Qui sono riportati, infatti, gli andamenti nel tempo delle forze esercitate da alcuni thruster sia in condizioni ideali (blu) che in presenza di un guasto sul quarto attuatore (rosso). Possiamo osservare che anche se l'unico thruster non funzionante è il quarto, f_4 non è l'unica variabile che modifica il suo comportamento, ma anzi tutte le forze in gioco si comportano in modo molto differente da quello previsto. Questo è probabilmente dovuto agli anelli di controllo presenti (descritti nel paragrafo 2.2) che, rilevando una risposta anomala da parte del veicolo, modificano l'azione di tutti gli attuatori. È chiaro, quindi, che una modellazione dei guasti come quella considerata finora, non è adatta per il nostro sistema, dal momento che per poter applicare la PCA, ogni guasto dovrebbe influire su un'unica variabile, eliminata la quale la struttura di



(a) SPE_1. (b) SPE_2.



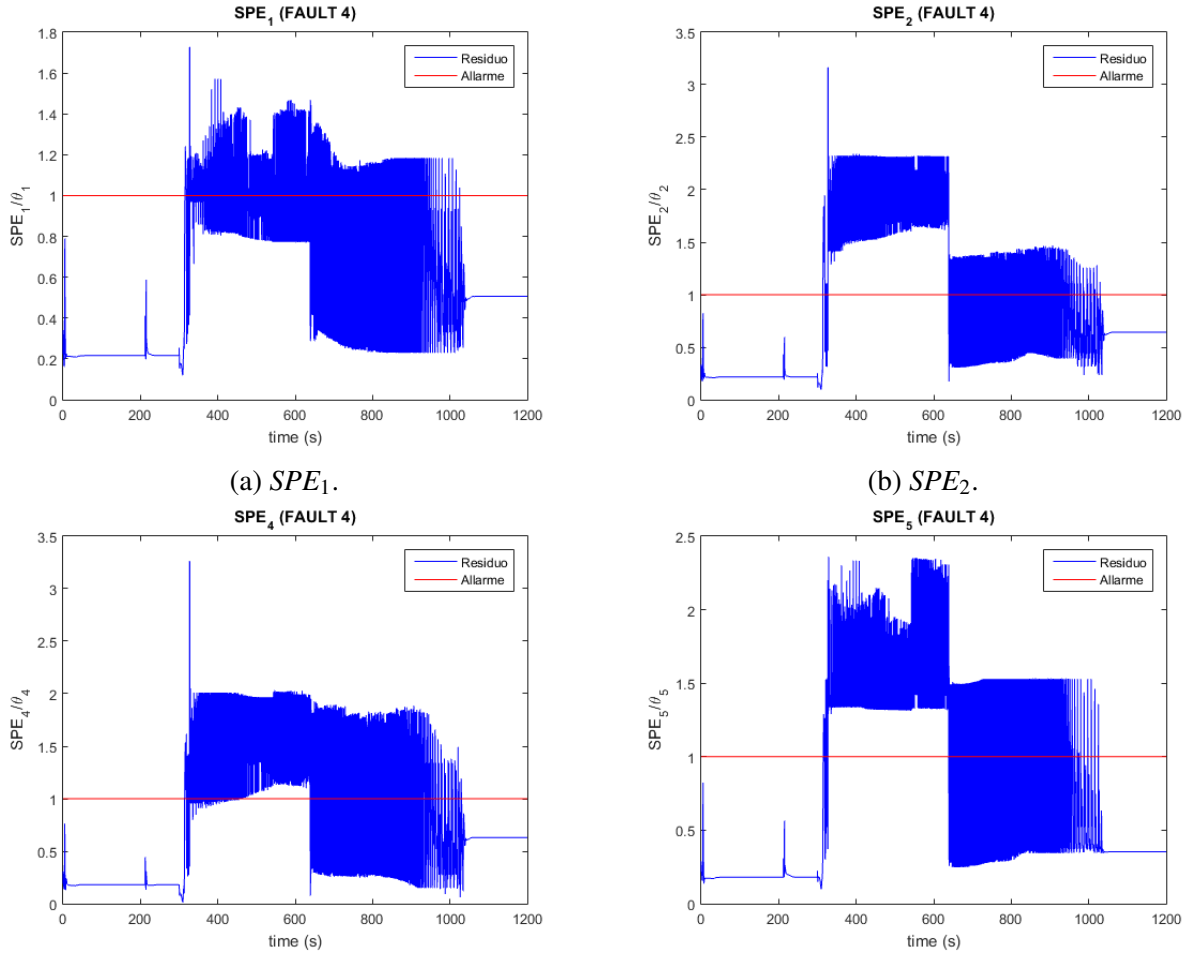
(c) SPE_4. (d) SPE_5.

Figura 3.23: Alcuni Residui Strutturati (blu) e la soglia del segnale d'allarme (rosso), ottenuti simulando un guasto sul thruster 1 a $t = 300\ s$.

correlazione dei dati dovrebbe essere mantenuta. Una situazione come quella realmente osservata, perciò, è equivalente alla presenza di un guasto su ogni attuatore.
Tutto questo porta a pensare che la strada percorsa fino a questo momento non possa essere applicata per risolvere il problema di Fault Isolation (FI), ma solo quello di Fault Detection (FD).

3.4 FD su V-Fides

Nel paragrafo precedente abbiamo supposto di conoscere direttamente le forze applicate da ogni thruster. Questa misura, se disponibile, rappresenterebbe il dato più informativo per monitorare lo stato degli attuatori. Sappiamo bene, però, che nella pratica il valore di tali forze non è noto ed è quindi necessario selezionare delle nuove variabili se vogliamo poter applicare l'algoritmo della PCA al sistema reale. Un'idea possibile è quella di considerare come vettore di stato, il vettore ν delle velocità lineari ed angolari in *body frame*.
Sulla base del dataset acquisito tramite simulazione del veicolo in assenza di guasti (vedi paragrafo 3.1), poniamo $l = 2$. Tale scelta del numero delle componenti principali non solo permette di catturare oltre il 99.9% della varianza del sistema, ma ci viene suggerita anche dal grafico in figura 3.26, nel quale è osservabile un "ginocchio" ben definito in corrispondenza del punto selezionato.



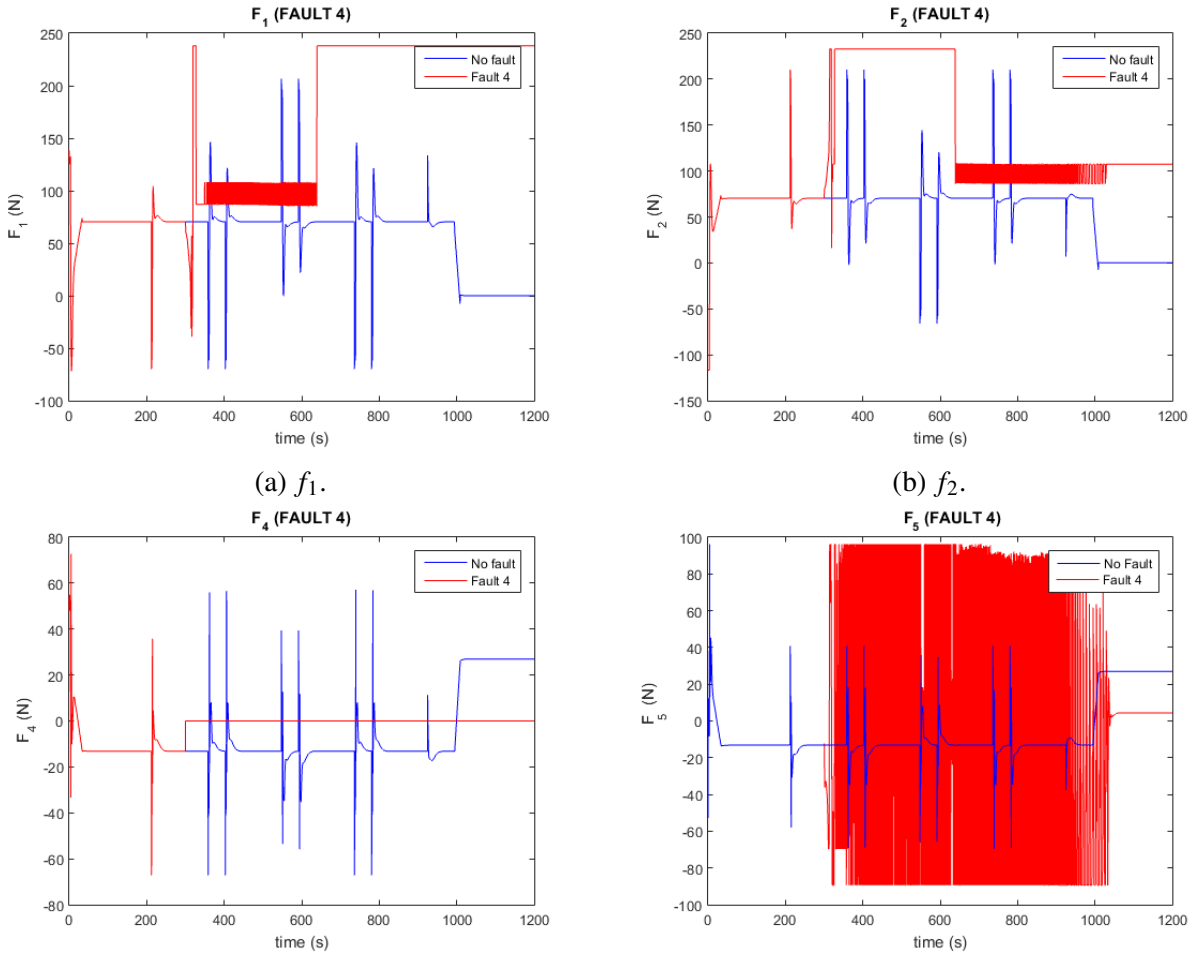
(a) SPE_1. (b) SPE_2.

(c) SPE_4. (d) SPE_5.

Figura 3.24: Alcuni Residui Strutturati (blu) e la soglia del segnale d'allarme (rosso) ottenuti simulando un guasto sul thruster 4 a $t = 300\ s$.

In questo caso, a differenza di quanto fatto nel paragrafo precedente, non sono stati elaborati dei modelli di fault. Questi, infatti, servono solo nel caso in cui si intenda procedere con l'isolamento del guasto, ma abbiamo già osservato che la versione standard della PCA non è in grado di effettuare tale operazione nemmeno avendo a disposizione il vettore dlle forze f, ovvero il dato maggiormente indicativo riguardo lo stato degli attuatori. Limitandoci allo studio della FD, procediamo con il calcolo della matrice $\tilde{C}$ di proiezione su RS, come mostrato nell'equazione 1.4.
A questo punto è possibile testare la validità dell'algoritmo attraverso delle simulazioni effettuate in assenza e in presenza di guasti sui thruster. In fugura 3.27 è riportato l'andamento nel tempo, durante le simulazioni, del residuo *SPE* scalato per una soglia θ. Tale soglia è stata determinata sulla base dei valori assunti dal residuo in condizioni nominali. È opportuno osservare che, durante i test, al residuo è stato applicato un filtro a media mobile per renderne il comportamento più *smooth*.

Possiamo notare che tutti i guasti sono stati rilevati correttamente, persino quello sul terzo attuatore che, come abbiamo già potuto osservare dal grafico in figura 3.9, ha una bassa influenza sul sitema dal momento che è poco utilizzato durante la survey.



(a) f_1. (b) f_2.

(c) f_4. (d) f_5.

Figura 3.25: Forze esercitate da alcuni thruster in assenza di guasti (blu) e in presenza di un guasto sul thruster 4 a $t = 300\ s$ (rosso).

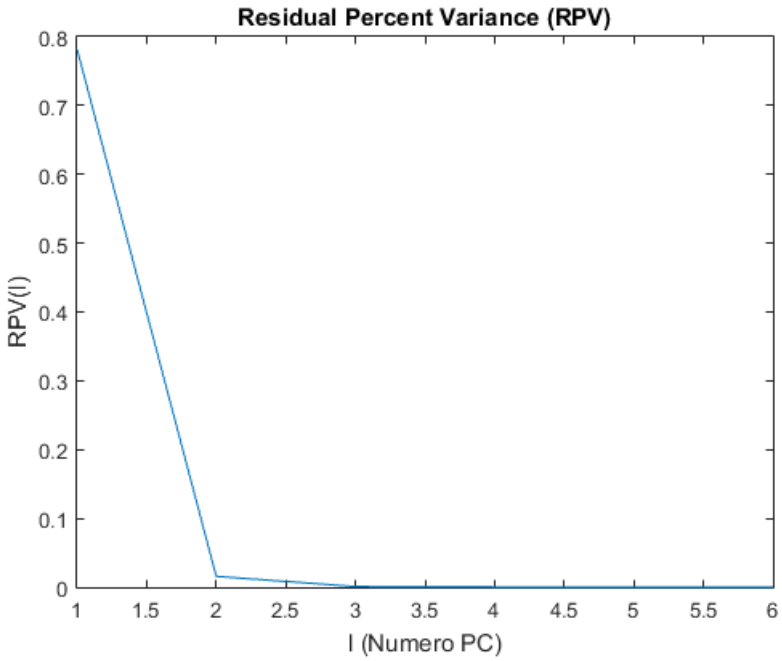
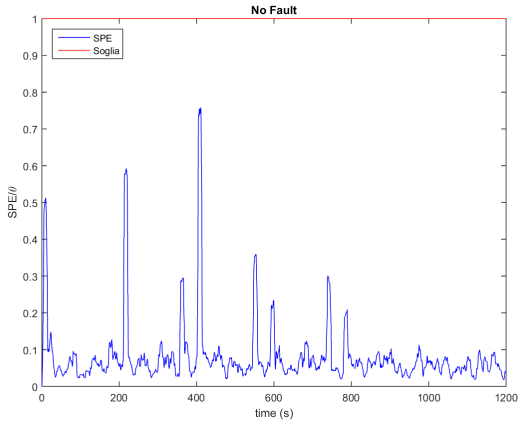
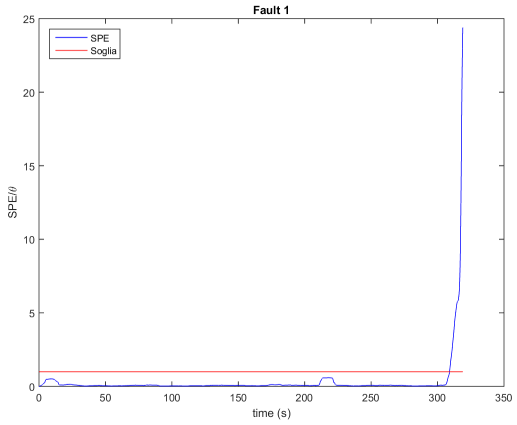


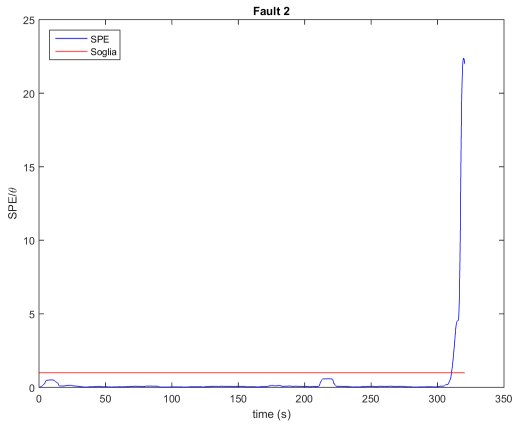
Figura 3.26: Residual Percent Variance del sistema con vettore di stato v.



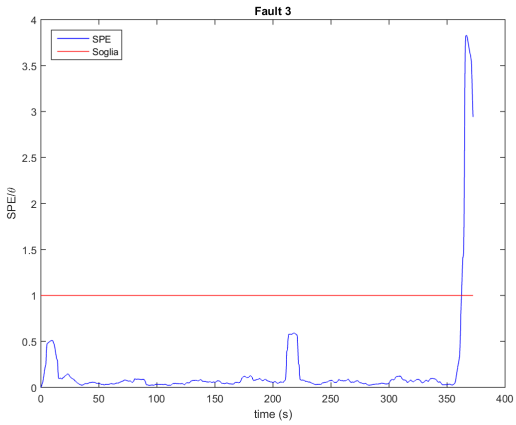
(a) *No fault*.



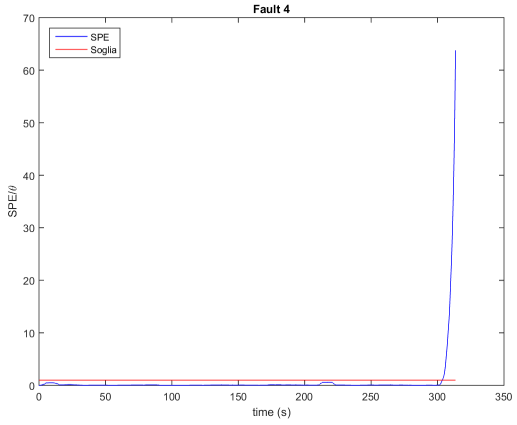
(b) *Fault 1*.



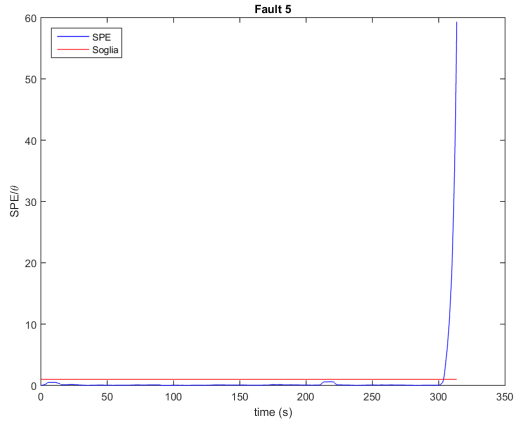
(c) *Fault 2*.



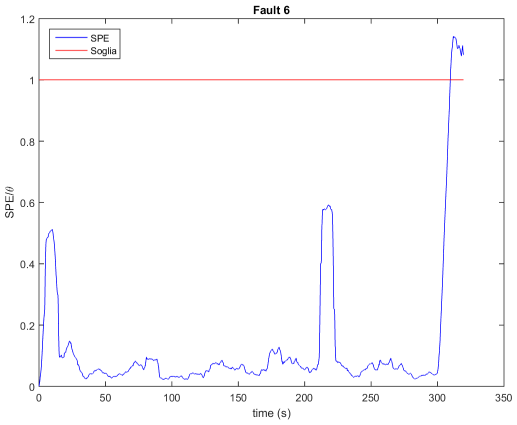
(d) *Fault 3*.



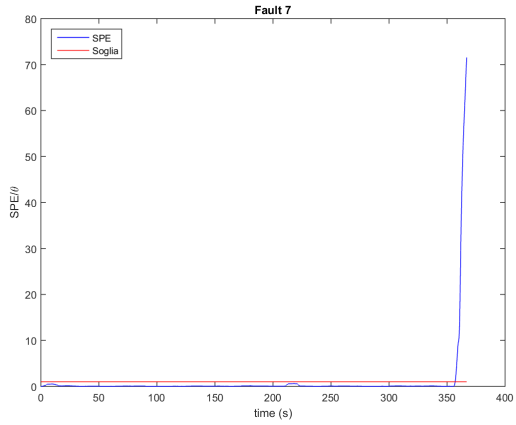
(e) *Fault 4*.



(f) *Fault 5*.



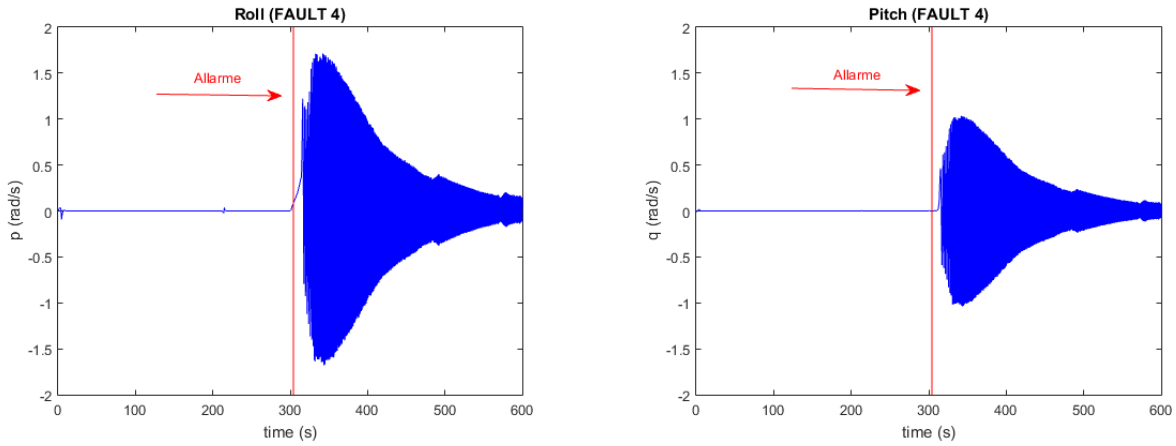
(g) *Fault 6*.



(h) *Fault 7*.

Figura 3.27: SPE/θ al variare del guasto. Il fault è stato simulato a partire dall'istante $t = 300$ *s* e ogni simulazione è stata interrota dopo un intervallo di 10 *s* nel quale il residuo si mantiene sempre al di sopra della soglia d'allarme.

È opportuno sottolineare come il modulo di FD implementato sia stato in grado di rilevare ogni guasto con un tempo di latenza molto breve, ovvero prima che questo porti il sistema all'instabilità. A titolo di esempio si può osservare il grafico in figura 3.28, dove sono mostrati sia l'andamento nel tempo delle velocità di *roll* e *pitch* in presenza di un guasto sul quarto thruster, sia l'istante nel quale il segnale d'allarme è stato generato.



(a) *Velocità di rollio (p).* (b) *Velocità di beccheggio (q).*

Figura 3.28: Velocità di roll e pitch (blu) in presenza di un guasto sul quarto attuatore (simulato a partire dall'istante $t = 300\ s$) e istante nel quale il segnale d'allarme (rosso) è stato generato.

3.5 Metodi Alternativi per FI

Abbiamo visto, quindi, come la tecnica della PCA riesca a risolvere il problema di FD in modo affidabile (rilevazioni corrette di guasti senza la generazione di falsi allarmi anche in presenza di rumore sulle misure) ed efficiente, consentendo così al sistema di rendersi conto di un'eventuale anomalia prima che il veicolo diventi instabile. Rimane ancora aperto il problema di come individuare il guasto e come poterlo, perciò, compensare, consentendo al veicolo di portare comunque a termine la missione.

Nel tentativo di risolvere questo problema sono state considerate varie soluzioni. Di seguito ne riporteremo due ovvero Fault Isolation tramite set di modelli PCA e tramite rete neurale. Nonostante solo il secondo metodo abbia consentito di ottenere dei risultati interessanti, riportiamo anche una descrizione del primo, in quanto questo è alla base della tecnica di FI che verrà presentata nel capitolo successivo.

3.5.1 Set di modelli PCA

Supponiamo di conoscere il sottospazio delle componenti principali (e quindi la matrice $\tilde{C}_i$ di proiezione nel sottospazio residuo) associato al comportamento del veicolo con thruster i-esimo danneggiato. In analogia a quanto fatto nel paragrafo precedente, è possiibile associare al modello considerato un residuo strutturato SPE_i definito nel modo seguente:

$$SPE_i = \left\|\tilde{C}_i x\right\|^2 \quad . \tag{3.1}$$

Monitornando *on-line* il valore del residuo, si ha un'indicazione di quanto il comportamento del veicolo sia coerente con quello previsto dal modello. Avremo, perciò, che bassi valori di SPE_i corrispondono ad un'alta probabilità che nel sistema si sia verificato il guasto i.
Ripetendo il ragionamento per tutti gli attuatori, avremo 7 differenti residui ognuno corrispondente ad un guasto specifico sul veicolo. Quando il modulo di FD segnala la presenza di un'anomalia, è sufficiente guardare quale dei residui risulta più piccolo e sotto una soglia opportuna. Tale residuo ci indicherà quale attuatore si è effettivamente rotto causando la generazione del segnale di allarme.
Per poter elaborare un modello PCA di un sistema, però, è necessario disporre di un dataset che ne descriva il comportamento nelle condizioni desiderate. Sappiamo, invece, che fisicamente non è possibile acquisire dati nelle condizioni richieste in quanto non possiamo eseguire una survey con un attuatore fuori uso solo per studiarne il comportamento. Fortunatamente lo stesso limite non è presente in fase di simulazione dove eventuali guasti sui thruster possono essere simulati senza incorrere in alcuna conseguenza dannosa. Perciò la parte di FI, a differenza di quella di FD, risulta fortemente dipendente dal modello matematico utilizzato, tanto che un modello non accurato del sistema potrebbe portare a risultati tutt'altro che affidabili.
A questo punto resta da spiegare come ottenere i dataset che permettono di costruire i modelli PCA del veicolo in presenza di guasti.
Si pongono innanzitutto due problemi: stabilire a che istante bisogna simulare l'insorgere del guasto e per quanto tempo acquisire i dati dalla rottura dell'attuatore. Dal momento che il risultato ideale sarebbe isolare il guasto proprio nello stesso momento in cui questo viene rilevato, un'idea potrebbe essere quella di interrompere la simulazione nell'istante in cui il segnale d'allarme è generato e considerare quindi solo i dati acquisiti dall'insorgere del guasto fino al termine della simulazione. Abbiamo già osservato nel paragrafo 3.2, come, infatti, l'assetto del veicolo tenda a diventare rapidamente instabile dopo la rottura di un attuatore, mentre il nostro obiettivo è di isolare il thruster prima che questa situazione si verifichi. Proseguire con l'acquisizione dei dati anche dopo che il guasto è stato rilevato, potrebbe, perciò, rischiare di portare solo un peggioramento delle prestazioni dell'algoritmo.
Per qunato riguarda la scelta degli istanti in cui simulare il danneggiamento di un motore il discorso risulta un po' più delicato. Il comportamento del veicolo, infatti, cambia notevolmente a seconda della manovra che questo sta effettuando ed è logico supporre che si osserveranno dinamiche differenti a seconda se il guasto viene simulato in corrispondenza di un tratto rettilineo o durante una curva. L'esempio più lampante di questo fenomeno sono gli attuatori 3 e 7 che, come è già stato spiegato precedentemente, vengono utilizzati (e quindi rilevati) solo quando il veicolo sta curvando.
Per la survey descritta nel paragrafo 3.1, le manovre ritenute di maggior interesse sono le seguenti:

1. Avanzamento lungo un tratto rettilineo a velocità massima;
2. Inizio di una curva in senso antiorario;
3. Conclusione di una curva in senso antiorario;
4. Inizio di una curva in senso orario;
5. Conclusione di una curva in senso orario.

Non sono stati presi in considerazione né il rallentamento e l'accelerazione del veicolo né lo stazionamentoo in una posizione nota, dal momento che, come spiegato nel paragrafo 3.1, la probabilità

che il guasto si verifichi in queste situazioni è molto bassa. In figura 3.29 sono indicati gli istanti che sono stati considerati per l'elaborazione dei dataset.

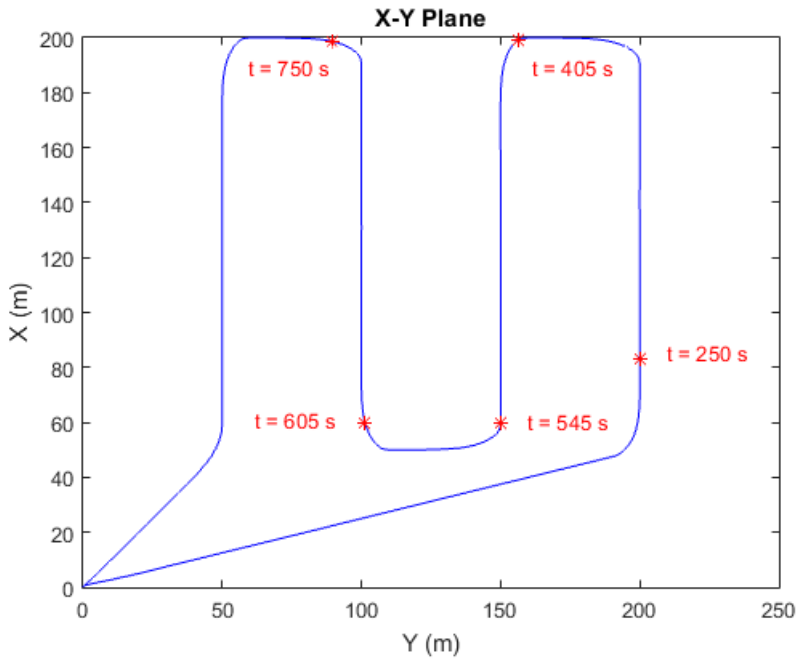


Figura 3.29: Traiettoria compiuta dal veicolo sul piano X-Y durante la survey con punti di interesse per la costruzione dei dataset per Fault Isolation.

Ricordiamo, inoltre, che i thruster 3 e 7 influenzano la dinamica del veicolo solo in corrispondenza delle curve, perciò per questi due attuatori non serve considerare tutti gli istanti presentati in figura 3.29. A questo punto possiamo definire per ogni thruster i, un vettore t_{F_i} contenente gli istanti ritenuti di maggior interesse per l'elaborazione del modello corrispondente. In particolare nel nostro caso avremo:

- $t_{F_i} = [250 \;\; 405 \;\; 545 \;\; 605 \;\; 750]^T \; s, \; \forall \, i \neq 3 \; \wedge \; i \neq 7$
- $t_{F_i} = [405 \;\; 545]^T \; s, \;$ per $i = 3 \; \vee \; i = 7$

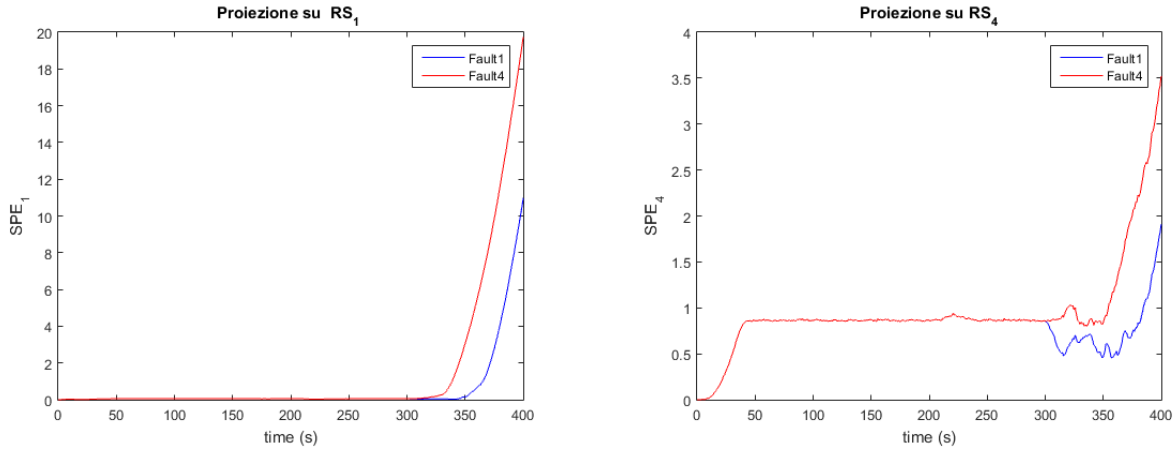
La procedura per l'elaborazione dei modelli PCA in caso di guasto, può essere schematizzata come segue:

Acquisizione Dataset per elaborazione modelli FI:

1. $i = 1$ (inizializzazione indice del thruster);
2. $j = 1$ (inizializzaione indice della manovra)
3. $l_i = length(t_{F_i})$ (calcolo numero di manovre per il guasto i);
4. Simulare il comportamento del veicolo durante la survey prevedendo la rottura del thruster i all'istante $t_{F_i}(j)$;

5. Interrompere la simulazione quando il modulo di FD rileva la presenza del guasto;
6. Memorizzare i dati acquisiti dall'istante $t_{F_i}(j)$ fino al termine della simulzaione;
7. Se $j < l_i \rightarrow j = j+1$ e tornare al passo 4;
8. Se $j = l_i \rightarrow$ elaborare un modello PCA associato al thruster i con i dati memorizzati;
9. Se $i < 7 \rightarrow i = i+1$ e tornare al passo 2;
10. Se $i = 7 \rightarrow$ l'algoritmo è terminato.

Purtroppo le simulazioni effettuate mostrano che la soluzione appena esposta non è in grado di isolare correttamente i guasti considerati, indipendentemente dal modo in cui viene scelto il vettore dei dati x. In particolare in figura 3.30 è mostrato l'andamento nel tempo dei residui SPE_1 e SPE_4 con $x = v$, sia in presenza di un guasto sul primo attuatore, che in presenza di un guasto sul quarto (in entrambi i casi si è supposto che il guasto agisca dall'istante $t = 300\ s$). Differentemente da quanto atteso, entrambi i residui risultano maggiori nel caso di un fault sul quarto attuatore. La tecnica testata non è quindi in grado di suggerire quale sia il guasto realmente avvenuto nel veicolo.



(a) SPE_1. (b) SPE_4.

Figura 3.30: SPE_1 e SPE_4 sia nel caso di guasto sul primo attuatore che nel caso di guasto sul quarto (in entrambi i casi il guasto è stato simulato a partire dall'istante $t = 300\ s$).

Questo fenomeno può probabilmente dipendere dal fatto che in presenza di guasti, il veicolo non può più essere approssimato attraverso un modello lineare. Nel paragrafo 2.2 è stato già spiegato, infatti, come gli anelli di controllo presenti non si limitino a dirigere il veicolo lungo la traiettoria desiderata, ma effettuino anche una parziale linearizzazione e disaccoppiamento del sistema, che però viene a ,ancare quando si ha la rottura di uno degli attuatori. Nel paragrafo 4.6 verrà mostrato come una modellazione differente della struttura di correlazione tra le variabili del sistema possa portare a risultati decisamente migliori.

3.5.2 FI con Reti Neurali

Le reti neurali artificiali sono algoritmi bio-ispirati nati per riprodurre attività tipiche del cervello umano come la percezione di immagini, il riconoscimento di forme, la comprensione del

linguaggio, il coordinamento senso-motorio, ecc.
Proprio come il cervello umano, una rete neurale artificiale è una struttura distribuita e massicciamente parallela, costituita da una connessione di elementi molto semplici, capace di apprendere e quindi di generalizzare (cioè produrre uscite in corrispondenza di ingressi non incontrati durante l'addestramento).
Anche se di recente introduzione, le reti neurali trovano valida applicazione in settori quali predizione,controllo e soprattutto classificazione e riconoscimento, portando spesso contributi significativi alla soluzione di problemi difficilmente trattabili con metodologie classiche. Proprio per questa loro versatilità nel risolvere problemi di classificazione, si è deciso di utilizzare le reti neurali come strumento per la FI. Una volta che il modulo di FD ha rilevato la presenza di un guasto, infatti, stabilire quale, tra un insieme di possibili anomalie, è in grado di spiegare meglio il comportamento del veicolo, non è altro che un problema di *pattern recognition* (riconoscimento di un insieme), ovvero un compito per il quale le reti neurali artificiali sono ampiamente utilizzate.
Le reti neurali possono essere distinte in due categorie principali, ovvero reti ad addestramento supervisionato e *self organizing feature maps* (SOFM). La differenza più significativa tra queste due tipologie, riguarda il loro processo di "apprendimento". L'addestramento delle prime consiste, infatti, nel presentare in ingresso alla rete un insieme di esempi (*training set*). La risposta fornita dalla rete per ogni esempio viene confrontata con la risposta desiderata, si valuta la differenza (errore) fra le due e, in base a tale differenza, si aggiustano i pesi. Questo processo viene ripetuto sull'intero training set finchè le uscite della rete producono un errore al di sotto di una soglia prestabilita. Le SOFM sono, invece, delle mappe bi-dimensionali di neuroni in grado di autoorganizzarsi. Con questo si intende che, dopo una fase di addestramento non supervisionato nella quale sono presentati alla rete degli ingressi opportuni, le posizioni spaziali dei gruppi di neuroni eccitati della mappa rappresentano una mappa topologica dei vettori di ingresso. Le relazioni di distanza nello spazio n-dimensionale degli ingressi sono, quindi, rappresentate approssimativamente da relazioni di distanza sulla rete neurale bi-dimensionale (figura 3.31).

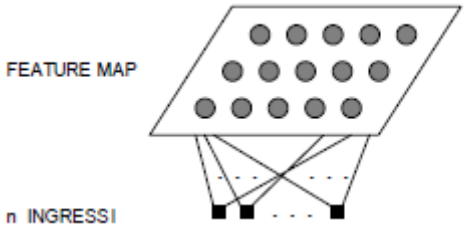


Figura 3.31: Rappresentazione di una *Self organizing feature map* generica.

Entrambe le tipologie sono adatte a risolvere problemi di riconoscimento e classificazione anche se con una differenza fondamentale: nel primo caso è l'utente che, durante la fase di addestramento, indica alla rete a quale classe appartiene ognuno degli ingressi forniti; nel secondo, invece, la mappa individua autonomamente degli insieme (*cluster*) e raggruppa ingressi "simili" all'interno dello stesso cluster.
Per quanto riguarda la FI dei guasti sul veicolo, si è deciso di utilizzare una rete ad addestramento supervisionato, anche se nulla vieterebbe di ricorrere alle SOFM oppure utilizzare una combinazione di entrambe.

Uno degli aspetti chiave per la generazione di una rete neurale, è la scelta di un training set opportuno che, nel nostro caso, è stato determinato con una procedura molto simile a quella già presentata nel paragrafo 3.5.1. L'unica differenza rispetto al caso precedente è che, stavolta per ogni vettore del dataset, dobbiamo predisporre anche un vettore di uscita $x_{out} \in \mathbb{R}^7$ costituito da tutti zeri ed un

unico 1 in posizione *i*, dove *i* indica il numero del thruster rotto.
Una volta applicato l'algoritmo, avremo un insieme di ingressi con relative uscite, che ci permetterà di procedere con l'addetramento supervisionato della rete. Quest'ultima è una rete stratificata, ovvero nella quale è possibile individuare degli strati di neuroni tali che ogni neurone è connesso con tutti quelli dello strato successivo, ma non esistono connessioni tra i neuroni all'interno dello stesso strato, né tra neuroni di strati non adiacenti. Nel nostro caso, in particolare, è possibile riconoscere tre strati, uno di ingresso, uno di uscita e uno intermedio detto strato nascosto. Dal momento che i segnali viaggiano dallo strato di ingresso verso quello di uscita, si parla di rete feedforward. In figura 3.32 è mostrata l'architettura di una rete generica con questa struttura.

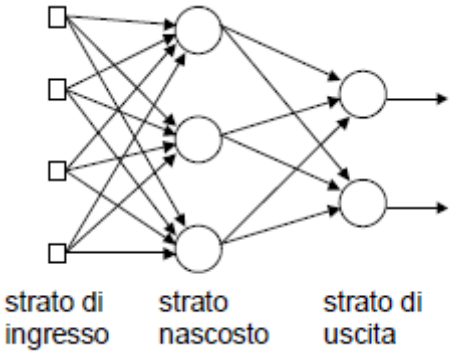


Figura 3.32: Architettura di una rete stratificata feedforward, con un solo strato nascosto.

La rete è stata realizzata mediante la *Neural Network Toolbox* di MATLAB e presenta rispettivamente 6 neuroni nello strato di ingresso, 10 in quello nascosto e 7 in quello di uscita.
Ogni uscita *i* avrà un valore compreso tra 0 e 1 e può essere vista come un differente classificatore continuo che indica la probabilità che nel sistema sia effettivamente avvenuto il guasto sul thruster *i*-esimo. In figura 3.33 sono riportati i risultati dell'analisi tramite *Receiver Operating Characteristics* (ROC) effettuata sui 7 classificatori, sia durante la fase di addestramento, che di validazione e verifica.

Osservando i grafici ROC, possiamo concludere che la rete neurale presenta delle buone prestazioni nella classificazione dei dati, dal momento che le curve sono molto simili a quella corrispondente al classificatore perfetto riportata in figura 3.34.

A questo punto non ci resta che testare la rete neurale implementandola direttamente sul modello simulink descritto nel paragrafo 2.3. In figura 3.35 sono riportate le uscite della rete, al variare del thruster rotto, nell'intervallo temporale che va dall'insorgere del guasto fino alla rilevazione dello stesso tramite PCA. Ogni guasto è stato simulato a partire dall'istante $t = 300\ [s]$.

Riportiamo adesso alcune considerazioni riguardo i grafici in figura 3.35. Osserviamo innanzitutto che, indipendentemente dal guasto, vi è sempre una fase iniziale, più o meno breve, in cui la terza uscita della rete neurale è quella con il valore più alto. Questo è probabilmente dovuto al fatto che l'effetto del guasto non si è ancora fatto sentire sul sistema e quindi la situazione più probabile, secondo la rete neurale, è che il thruster rotto sia il settimo, data la sua scarsa influenza sulla survey (come evidenziato nel paragrafo 3.2.5). In ogni caso, però, il basso valore dell'uscita dimostra che, seppur sia più probabile delle altre, rimane scarsa la probabilità che la rottura del settimo attuatore sia effettivamente avvenuta. Inoltre, mentre tutti i guasti vengono rilevati e isolati dopo pochi secondi, il blocco del terzo e del settimo motore viene risentito solo quando il veicolo inizia a curvare (vedi figura 3.4). La ragione di ciò è dovuta al fatto che, come gia spiegato nei

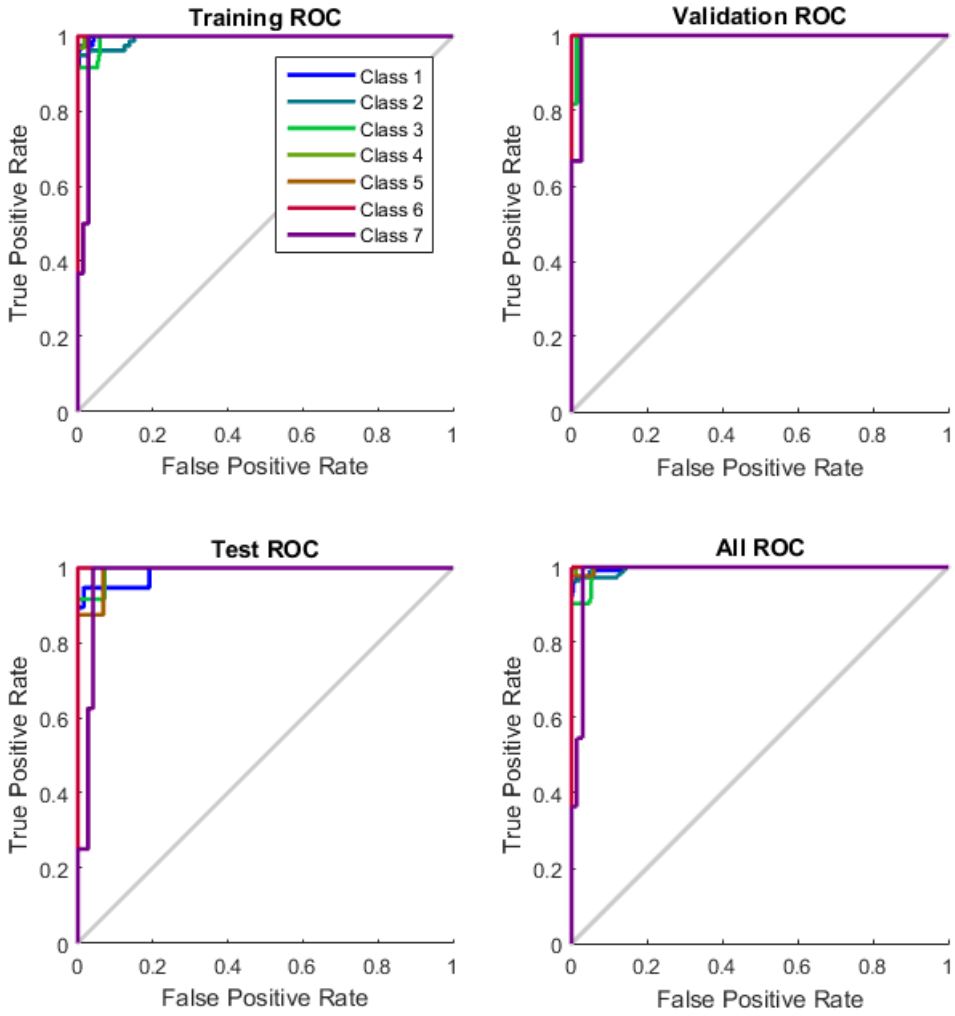


Figura 3.33: Grafici ROC dei 7 classificatori ottenuti utilizzando la rete neurale per FI.

paragrafi 3.2.2 e 3.2.5, i due thruster agiscono solo durante le curve, esercitando forze pressoché nulle durante il resto della survey. Ad ogni modo, quando il modulo di FD rileva la presenza di un'anomalia nel veicolo, la rete neurale è sempre in grado di distinguerne l'origine ed isolare correttamente il guasto corrispondente.

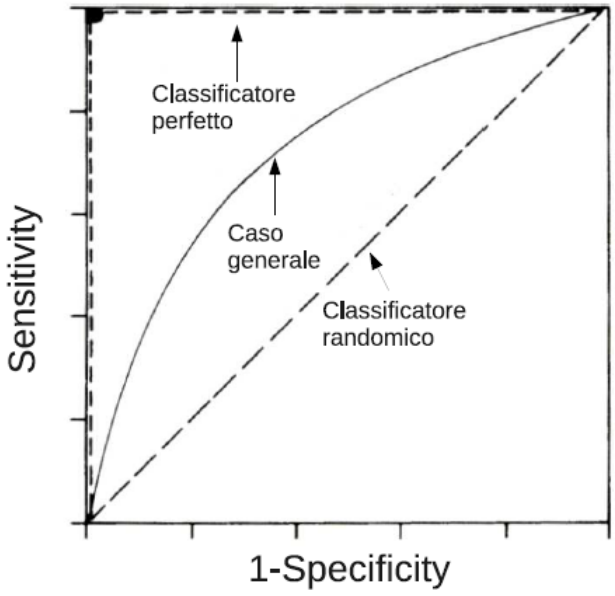
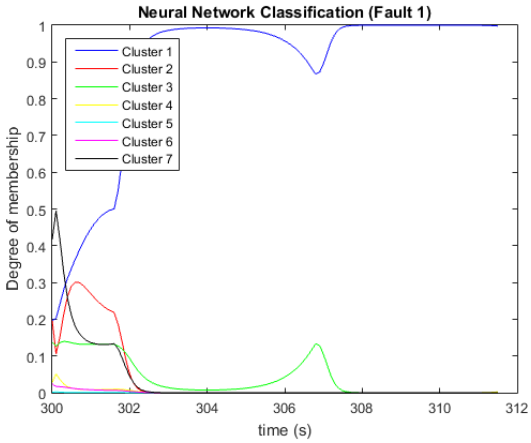
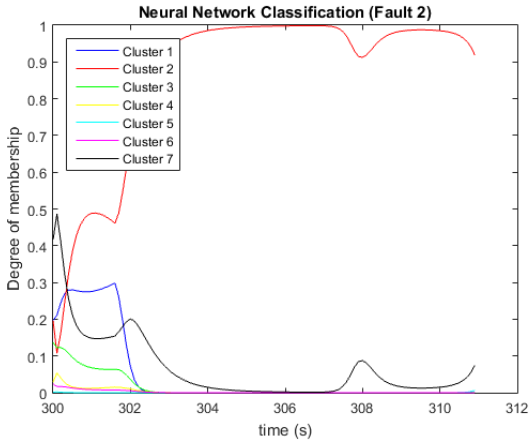


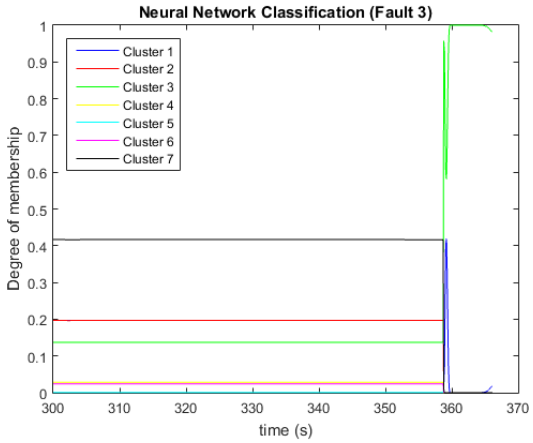
Figura 3.34: Grafici ROC per un classificatore generico, casuale o perfetto.



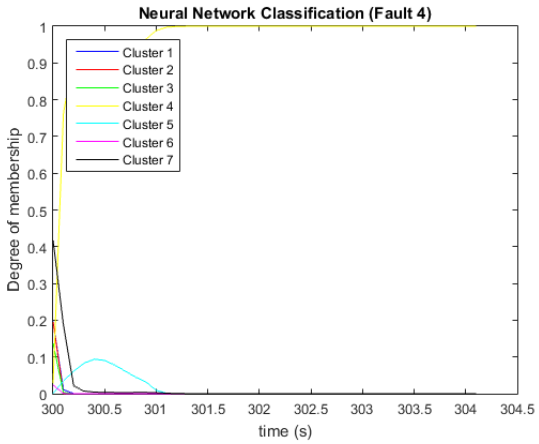
(a) *Fault 1*.



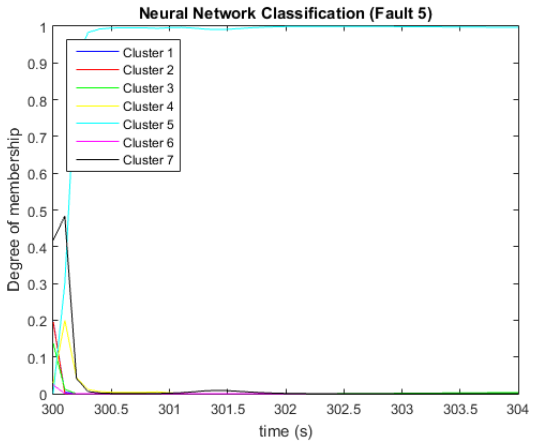
(b) *Fault 2*.



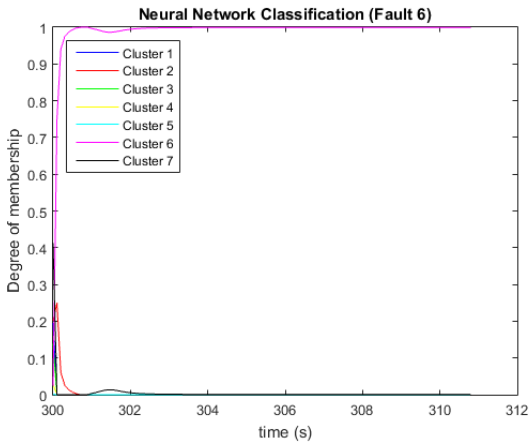
(c) *Fault 3*.



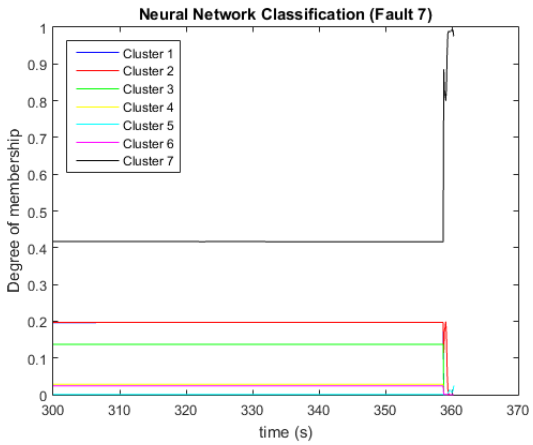
(d) *Fault 4*.



(e) *Fault 5*.



(f) *Fault 6*.



(g) *Fault 7*.

Figura 3.35: Uscita della rete neurale per FI durante l'intervallo temporale che va dall'insorgere del guasto ($t = 300\ s$) fino alla rilevazione dello stesso tramite PCA.

CAPITOLO 4

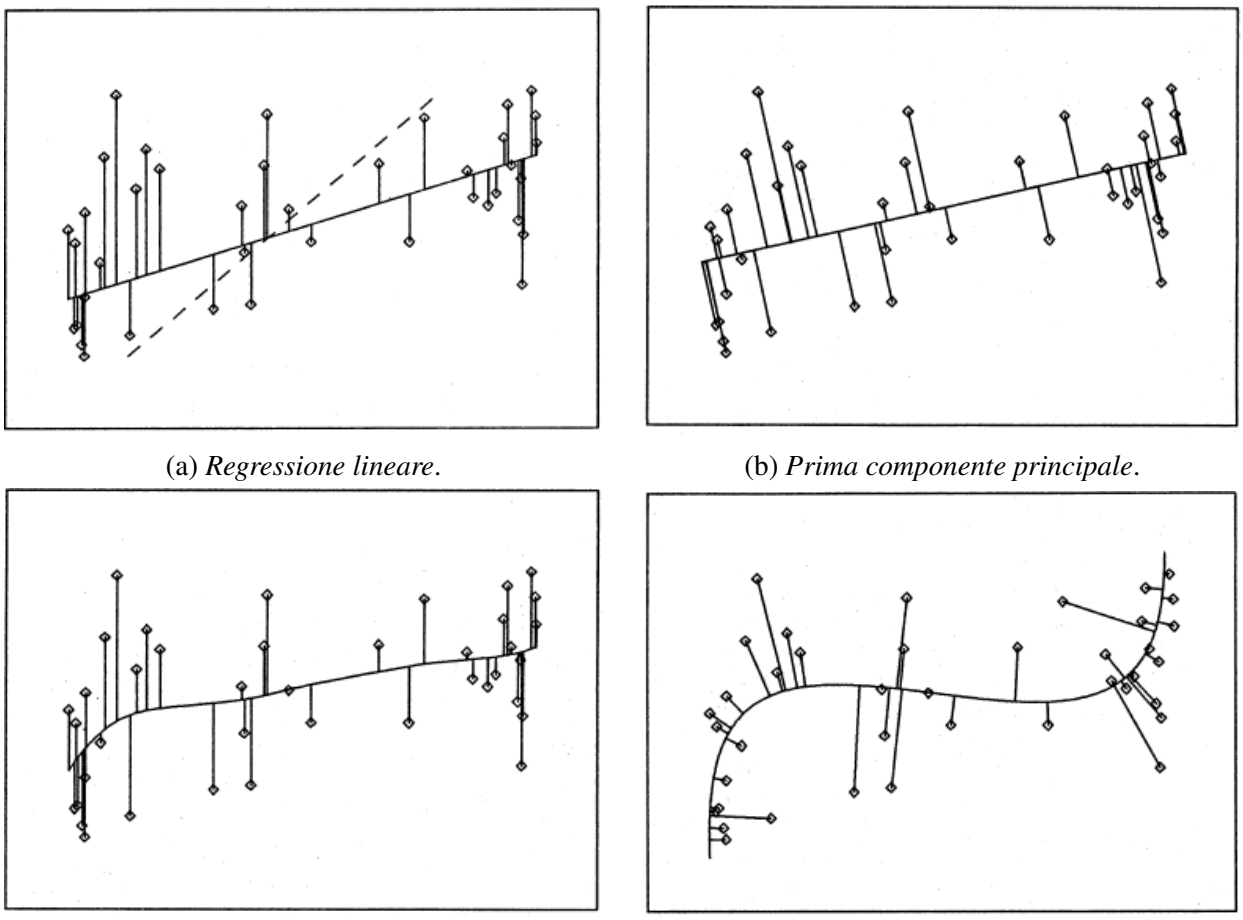
NON LINEAR PCA

Nel capitolo precedente abbiamo visto come la PCA possa essere utilizzata per rilevare i guasti sugli attuatori del veicolo marino sovrattuato V-Fides, ma è stato anche messo in evidenza come questa tecnica non sia in grado identificare quale sia il thruster effettivamente rotto. Una spiegazione possibile del perché ciò si verifichi è che la PCA è un metodo lineare a differenza del processo in esame. Finché nel veicolo non sono presenti anomalie e tutti gli attuatori stanno funzionando correttamente, infatti, il comportamento del sistema può essere approssimato tramite un modello lineare, ma quando un motore si rompe, allora gli effetti delle non linearità non possono più essere trascurati.
In questo capitolo verrà presentata una possibile estensione non lineare dell'analisi delle componenti principali e verrà messo in evidenza come questa tecnica possa essere applicata per risolvere il probema di FDI sul veicolo.
Per una trattazione più approfondita dell'argomento, fare riferimento a [7] e [3].

4.1 Principal Curves Analysis

Consideriamo un dataset costituito da N misure di due variabili scalari x e y. È possibile rappresentare tali coppie di variabili come N punti di un piano, come illustrato in figura 4.1a. In questo caso, ma anche in situazioni più complesse con un numero maggiore di variabili, viene spontaneo cercare di descrivere il sistema in un modo più compatto, ovvero cercando di fornire il maggior numero di informazioni possibili utilizzando il minor numero di dati. Questa "sintesi" del dataset può essere fatta in molti modi in base agli obiettivi della nostra analisi. Il metodo più banale per farlo è la determinazione del valor medio, il quale individua il centro della nuvola di punti, senza però fornire alcuna indicazione sulla relazione tra le due variabili. Per sopperire a questa mancanza, in genere si suddivide le variabili in gioco in due categorie: *variabili indipendenti* e *variabili dipendenti*. L'obiettivo diventa, perciò, individuare una regola che permetta di predire la risposta del sistema (variabili dipendenti) sulla base dei dati a nostra disposizione (variabili indipendenti). A tale scopo si utilizzano le cosiddette tecniche di regressione, il cui esempio più semplice è la regressione lineare. Quest'ultima modella la variabile di risposta y come una funzione lineare di x. Tale funzione viene solitamente stimata attraverso il metodo dei minimi quadrati, che nel nostro caso è equivalente a trovare la linea retta che minimizza la somma dei quadrati delle deviazioni verticali (come mostrato in figura 4.1a). Quando una relazione lineare non è in grado di rappresentare in modo accurato il comportamento delle variabili considerate, allora è possibile procedere con le tecniche di regressione non lineare. Queste, nel nostro caso, corrispondono a determinare quale, tra una famiglia di funzioni parametriche predefinite, sia in grado di minimizzare le deviazioni verticali (figura 4.1c).
È opportuno osservare che, indipendentemente dal metodo utilizzato, le tecniche di regressione richiedono sempre che i dati vengano divisi in un dataset di ingresso e in uno di risposta.
In molte situazioni, però, questo potrebbe causare dei problemi. La linea tratteggiata in figura 4.1a mostra il risultato che si otterebbe invertendo il ruolo di x e y durante la procedura di regressione lineare. È chiaro che il risultato è molto diverso da quello ottenuto precedentemente (linea continua), quindi scegliere in modo casuale quale sia la variabile dipendente, potrebbe portare ad una rappresentazione poco significativa del dataset in esame. Spesso, però, non vi è alcuna indicazione

di quale sia la variabile di risposta e in tal caso, perciò, si preferirebbe trattare tutte le variabili in modo simmetrico. Proprio per questo motivo si può ricorrere al calcolo della prima componente principale del sistema, ovvero la determinazione della linea retta in grado di minimizzare le deviazioni ortogonali (figura 4.1b).
Allo stesso modo in cui la regressione lineare può essere generalizzata attraverso le regressioni non lineari, così si può procedere anche per lo studio delle componenti principali. Al posto di determinare una linea retta, perciò, andremo a cercare una curva liscia in grado di minimizzare le distanze ortogonali tra le variabili, che quindi saranno anche in questo caso trattate in modo simmetrico senza alcuna distinzione tra variabili di ingresso e uscita (figura 4.1d).
La curva ottenuta è detta prima curva principale del sistema e gioca lo stesso ruolo svolto dalla prima componente principale definita nel capitolo 1. Come nel caso lineare, anche in questo, una volta determinata la prima curva principale, è possibile procedere con il calcolo delle curve principali successive.
Nei pragrafi successivi verrà fornita una definizione formale delle curve principali e verrà mostrato come queste possano essere calcolate e utilizzate durante lo studio di un sistema. Prima di procedere con la trattazione, però, verrà data una breve descrizione di cosa si intenda per curve lisce unidimensionali.



(a) *Regressione lineare*. (b) *Prima componente principale*.

(c) *Regressione non lineare*. (d) *Prima curva principale*.

Figura 4.1: Modellazione di un dataset costituito da coppie di variabili x e y, mediante linee rette o curve.

4.2 Curve 1-D

Una curva unidimensionale in uno spazio p-dimensionale è un vettore $f(\lambda)$ di p funzioni di una singola variabile scalare λ. Formalmente avremo, quindi, che $f(\lambda) : \mathbb{R} \to \mathbb{R}^p$ con $\lambda \in \mathbb{R}$. Le componenti del vettore $f(\lambda)$ sono chiamate funzioni coordinate, mentre λ fornisce un ordinamento lungo la curva. Se le funzioni coordinate sono lisce (infinitamente derivabili), allora f è per definizione una curva liscia. Applicando una qualsiasi trasformazione monotona a λ (e modificando di conseguenza le funzioni coordinate), la curva f non cambia, l'unica differenza si avrà nel modo in cui questa è parametrizzata. Per ogni curva esiste una parametrizzazione naturale in termini di lunghezza dell'arco. Dati due punti $f(\lambda_0$ e $f(\lambda_1)$ sulla curva f, la lunghezza dell'arco che li connete può essere calcolata come

$$l = \int_{\lambda_0}^{\lambda_1} \|f'(z)\| dz \ . \tag{4.1}$$

Il vettore $f'(\lambda)$ è tangente alla curva nel punto individuato da λ ed è spesso chiamato *vettore velocità in* λ. Una curva per la quale $\|f'\| = 1$ è detta curva parametrizzata a velocità unitaria. Ogni curva liscia può essere sempre riparametrizzata per essere resa a velocità unitaria. Se $v \in \mathbb{R}^p$ è un vettore unitario, allora avremo che

$$f(\lambda) = v_0 + \lambda v \tag{4.2}$$

con $v_0 \in \mathbb{R}^p$, è una linea retta a velocità unitaria.
Il vettore $f''(\lambda)$ è detto accelerazione della curva in λ e, per una curva a velocità unitaria, è facile verificare che risulta ortogonale al vattore tangente. In questo caso $f''/\|f''\|$ è chiamato normale principale alla curva in λ.
Si può dimostrare che esiste un unico cerchio a velocità unitaria, appartenente al piano generato dai vettori $f(\lambda)$ e $f'(\lambda)$, passante per il punto $f(\lambda)$ e avente in λ la stessa velocita e accelerazione della curva (figura 4.2).

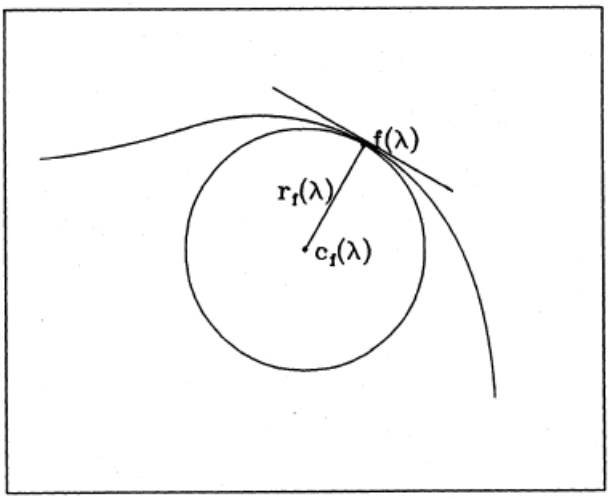


Figura 4.2: Il raggio di curvatura di una curva è il raggio del cerchio tangente alla curva e con stessa accelerazione e velocità.

Il raggio $r_f(\lambda)$ di questo cerchio è detto raggio di curvatura della curva f in λ. È possibile mostrare che

$$r_f(\lambda) = \frac{1}{\|f''(\lambda)\|} \quad . \tag{4.3}$$

Il centro $c_f(\lambda)$ del cerchio è chiamato, invece, centro di curvatura di f in λ.

4.3 Curve Principali

Una curva principale è una curva liscia unidimensionale che passa attraverso il "mezzo" di un dataset n-dimensionale. La sua forma è determinata dalla struttura dei dati dati e ne fornisce una rappresentazione non lineare e sintetica.

Sia $x \in \mathbb{R}^n$ un vettore casuale continuo con una distribuzione di probabilità nota p. Si dice che f è una curva principale di p se

$$E(x|\lambda_f(x) = \lambda) = f(\lambda) \tag{4.4}$$

dove con $\lambda_f : \mathbb{R}^n \to \mathbb{R}$ si intende l'indice di proiezione definito come

$$\lambda_f(x) = Sup_\lambda\{\lambda : \|x - f(\lambda)\| = inf_\mu\|x - f(\mu)\|\} \quad . \tag{4.5}$$

L'indice di proiezione $\lambda_f(x)$ del vettore x non è altro che il valore di λ in corrispondenza del quale si ha il punto della curva più vicino a x. Se sulla curva ci sono più punti $f(\lambda)$ equidistanti da x e che ne minimizzano la distanza, allora viene scelto come indice il valore di λ più grande tra questi.

Consideriamo a questo punto, un dataset $X \in R^{N\times n}$, costituito da N misure di n variabili e chiamiamo $f_1(\lambda)$ la prima curva principale di X. Supponiamo che la curva sia stata parametrizzata in modo naturale e che λ ne indichi quindi la lunghezza dell'arco.
Ad ogni elemento generico $x \in \mathbb{R}^n$ di X, è possibile associare un punto $f_1(\lambda)$ sulla curva principale e quindi un indice di lunghezza corrispondente λ. Il vettore $t_1 \in \mathbb{R}^N$ costituito dagli indici associati a tutti gli elementi di X, è detto *non linear principal component score vector*.
Matematicamente avremo che

$$\begin{aligned} t_1 &= g_1(X) \\ X &= f_1(t_1) + E_1 \end{aligned} \tag{4.6}$$

dove $E_1 \in \mathbb{R}^{N\times n}$ è la matrice residua. Come nel caso delle componenti principali lineari, anche per le curve principali è possibile che la prima non basti a descrivere il comportamento del sistema in modo esaustivo. In questa situazione è necessario quindi, calcolare le curve principali successive. Ognuna di queste viene determinata ripetendo la procedura indicata finora, ma utilizzando come dataset non la matrice X, bensì la matrice residua della curva principale precedente:

$$\begin{aligned} t_i &= g_i(E_{i-1}) \\ E_{i-1} &= f_i(t_i) + E_i \ . \end{aligned} \tag{4.7}$$

In modo compatto possiamo scrivere:

$$T = G(X) \tag{4.8}$$

$$X = F(T) + E_l \tag{4.9}$$

dove $T = [t_1, t_2, \ldots, t_l] \in \mathbb{R}^{N \times l}$ è la *non linear principal score matrix*, $E_l \in \mathbb{R}^{N \times n}$ è la matrice residua finale e l è il numero di componenti principali non lineari selezionato per rappresentare il sistema. Le funzioni non lineari $G(\cdot)$ e $F(\cdot)$ sono chiamate *non linear principal loading functions* e permettono rispettivamente di calcolare la score matrix T e di ricostruire la matrice dei dati $\hat{X}$:

$$\hat{X} = F(T) \ . \tag{4.10}$$

Se avessimo a disposizione le due funzioni $F(\cdot)$ e $G(\cdot)$, potremmo attuare una tecnica per la FD simile a quella già vista per i sistemi lineari e che può essere schematizzata nel modo seguente:

FD con Principal Curves Analysis:

ON-LINE:

1. Misurare, ad ogni istante di campionamento il vettore x costituito dalle variabili del sistema;
2. Calcolo dello score vector $\rightarrow\ t = G(x)$;
3. Ricostruzione del vettore dei dati $\rightarrow\ \hat{x} = F(t)$;
4. Errore di ricostruzione $\rightarrow\ e = x - \hat{x}$;
5. Calcolo del residuo SPE $\rightarrow\ r = \|e\|^2$;
6. Detta θ una soglia fissata in modo opportuno:
   - Se $r < \theta \rightarrow$ Nessun fault rilevato
   - Se $r > \theta \rightarrow$ Fault rilevato

Rimane aperto il problema di come individuare le funzioni $F(\cdot)$ e $G(\cdot)$.
In [7] è presentato un algoritmo iterativo che permette di individuare le curve principali di un sistema. L'algoritmo proposto, però, non riesce a determinare un modello per le curve, ma si limita a calcolare la score matrix T associata ad un dataset X. Nel paragrafo successivo è descritta una possibile soluzione al problema.

4.4 NLPCA con Reti Neurali

Supponiamo di avere un dataset X ottenuto misurando le variabili di interesse di un sistema in condizioni operative. Applicando l'algoritmo iterativo in [7], è possibile calcolare la score matrix T associata a X. Ci poniamo a questo punto il problema di trovare le funzioni non lineari $F(\cdot)$ e $G(\cdot)$, tali da soddisfare le equazioni 4.8 e 4.9.
Dal momento che non abbiamo alcun suggerimento sulla struttura delle due funzioni e visto che esistono tante possibili funzioni quanti sono i tipi di sistemi, non sembra opportuno ricorrere all'utilizzo di una tecnica di regressione standard. Un approccio che, invece, risulta particolarmente adeguato allo scopo, è quello che prevede l'applicazione di reti neurali e che è stato presentato in [3]. Tale tecnica, che prende il nome di Non Linear Principal Component Analysis (NLPCA),

necessita di due modelli, rappresentati rispettivamente da dua reti neurali distinte. La prima mappa il dataset n-dmensionale in uno l-dimensionale; la seconda, invece, effettua l'operazione inversa. È stato dimostrato che le reti neurali godono della proprietà di approssimatori universali, intendendo con questo che una mappa con un solo strato nascosto (costituito da elementi con funzione di attivazione sigmoidale) è in grado di approssimare una qualsiasi funzione continua con accuratezza arbitrariamente desiderata. In realtà, non solo la funzione stessa, ma anche tutte le sue derivate posssono essere approssimate attraverso un rete di questo tipo. Le dimostrazioni di queste proprietà possono essere trovate in [8].
In figura 4.3 è mostrata l'architettura delle due reti neurali utilizzate per approssimare le funzioni $F(\cdot)$ e $G(\cdot)$.

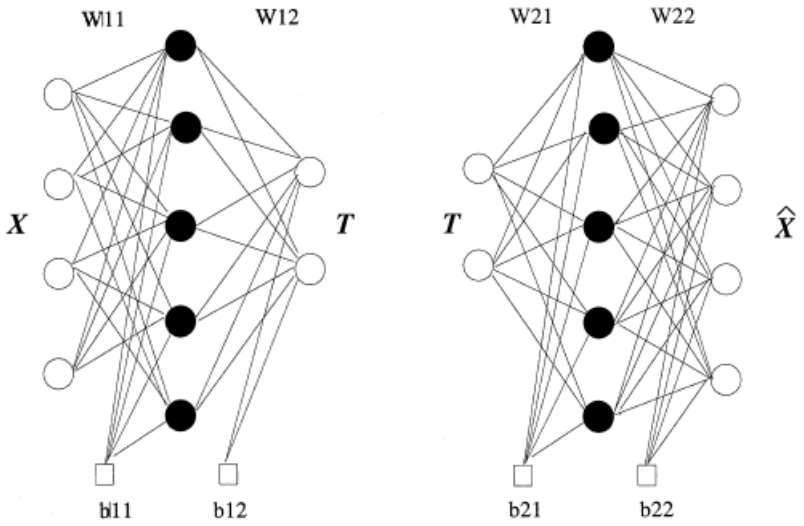


Figura 4.3: Reti neurali per NLPCA

Si tratta di due reti stratificate feedforward con tre strati ciascuna. Le equazioni 4.8 e 4.10 possono essere riscritte rispettivamente come

$$T = G(X) = W_{12}\sigma(W_{11}X + b_{11}) + b_{12} \tag{4.11}$$

$$\hat{X} = F(T) = W_{22}\sigma(W_{21}T + b_{21}) + b_{22} \tag{4.12}$$

dove con W_{ij} si è indicata la matrice dei pesi delle connessioni tra lo strato j e $j+1$ della rete i, mentre con b_{ij} si è indicato il bias corrispondente. Come funzione di attivazione dei neuroni dello strato nascosto si è scelta la funzione simmetrica sigmoidea $\sigma(\cdot)$ (figura 4.4), in modo da introdurre un comportamento non lineare nella rete. Come funzione di trasferimento degli strati di ingresso e uscita si è utilizzata, invece, una semplice funzione lineare.

Dal momento che lo strato di uscita della prima rete coincide con quello di ingresso della seconda, lo schema in figura 4.3 può essere visto come un'unica rete neurale con 5 strati (figura 4.5), costituita da due parti che possono essere addestrate separatamente.

La rete neurale in figura 4.5 è autoassociativa dal momento che pattern di ingresso e di uscita coincidono. Infatti l'uscita della rete altro non è che una ricostruzione dei dati in ingresso.

L'addestramento delle due reti in figura 4.3 è di tipo supervisionato (già descritto nel paragrafo 3.5.2) ed in entrambi i casi il training set è costituito dai dataset X e T, che invertono i ruoli di

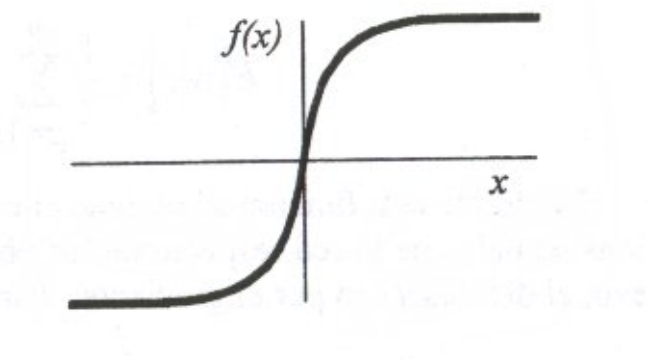


Figura 4.4: Funzione sigmoidea simmetrica.

ingresso e uscita a seconda della rete.

4.5 NLPCA su V-Fides

Una volta descritta la procedura per risolvere il problema di FD tramite NLPCA, non ci resta che applicare l'algoritmo al veicolo subacqueo in esame. Lo scenario è sempre quello presentato nel paragrafo 3.1, anche se a differenza di quanto fatto per l'applicazione della PCA standard, in questo caso si è scelto come vettore dei dati un vettore di stato aumentato $x = [u_{cmd}^T \ \nu^T]^T \in \mathbb{R}^{13}$, dove ricordiamo che con $u_{cmd} \in \mathbb{R}^7$ si è indicato i vettore dei comandi in ingresso agli attuatori, mentre con $\nu \in \mathbb{R}^6$ si è indicato il vettore delle velocità lineari e angolari del veicolo in *body frame*.
Seguendo le indicazioni fornite nel paragrafo 3.1, è possibile, quindi, acquisire il dataset $X \in$

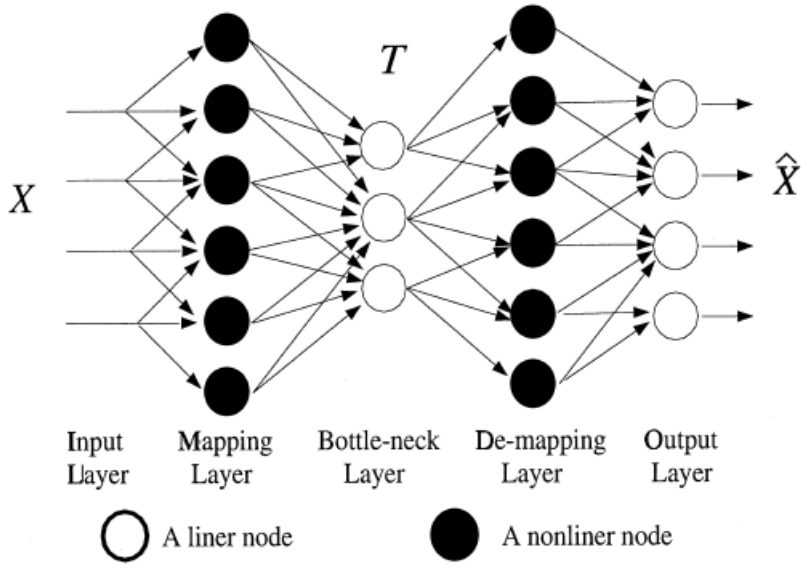


Figura 4.5: Architettura di una rete neurale autoassociativa a 5 strati.

$\mathbb{R}^{9000\times 13}$, sulla base del quale calcolare la matrice di score T, tramite l'applicazione dell'algoritmo in [7].
Dal momento che non esistono dei criteri che permettano di determinare il numero ottimale l di curve principali da utilizzare per l'applicazione della NLPCA, è necessario procedere per tentativi, secondo lo schema seguente:

1. Fissare una soglia critica θ_E;
2. Inizializzare $l = 1$;
3. Calcolare la l-esima curva principale;
4. Calcolare la matrice residua E_l dall'equazione 4.9;
5. Se $\|E_l\| > \theta_E \ \rightarrow l = l + 1$:
   - Se $l = 13 \ \rightarrow$ fissare una nuova soglia più grande della precedente e tornare al passo 2;
   - Se $l < 13 \ \rightarrow$ tornare al passo 3;
6. Se $\|E_l\| < \theta_E \ \rightarrow$ **return** l.

In realtà questa procedura, che può risultare molto lunga ed elaboriosa, può essere evitata grazie alla considerazione seguente. Nel capitolo 3, è stato mostrato come la PCA permetta di risolvere il problema di FD sul veicolo V-Fides, nonostante la dinamica del sistema sia non lineare; questo è probabilmente dovuto al fatto che, in assenza di guasti, il comportamento del veicolo non risente eccessivamente delle non linearità e può essere approssimato tramite un modello dinamico lineare. Quindi è logico supporre che le curve principali del sistema abbiano un andamento molto simile a quello delle rette rappresentanti le componenti principali. Il numero l può essere scelto, perciò, attraverso gli stessi criteri utilizzati per la PCA e riportati nel paragrafo 1.5. Vedremo, infatti, che il vero vantaggio portato dall'applicazione della NLPCA, si avrà per quanto riguarda la soluzione del problema di FI, mentre nel caso della FD i risultati saranno molto simili a quelli osservati per la PCA.

In figura 4.6 è mostrato l'andamento dell'indice RPV (equ. 1.21) utilizzando il dataset descritto sopra. A differenza dei grafici nelle figure 3.21 e 3.26, in questo caso non vi è un unico punto ben definito al quale corrisponde un "ginocchio", ma piuttosto ci sono due valori di l ($l = 5$ e $l = 6$) in corrispondenza dei quali la curva sembra "flettersi". Di questi due candidati possibili, si è preferito il più grande dal momento che la scelta dei primi 6 autovalori permette di catturare oltre il 99% della varianza della matrice dei dati (come è possibile osservare dalla tabella 4.1).

Una volta definito il numero di curve principali, possiamo procedere con il calcolo della matrice $T \in \mathbb{R}^{9000\times 6}$. A questo punto disponiamo di tutti gli elementi per la realizzazione delle due mappe in figura 4.3, tali da approssimare le funzioni $F(\cdot)$ e $G(\cdot)$ delle equazioni 4.8 e 4.10.
Le reti neurali sono state progettate utilizzando la *Neural Fitting Toolbox* di MATLAB. Questa fornisce un utile strumento per elaborare reti neurali in grado di approssimare una qualsiasi funzione ingresso-uscita, sulla base di due dataset (input e target). L'architettura delle mappe utilizzate per risolvere il problema di fitting (figura 4.7) è proprio quella desiderata per l'applicazione della NLPCA, ovvero con un unico strato nascosto, costituito da elementi a funzione di attivazione simmetrica sigmoidea e uno strato di uscita con neuroni a funzione di attivazione lineare. Entrambe le reti sono state addestrate con l'algoritmo del gradiente coniugato scalato.

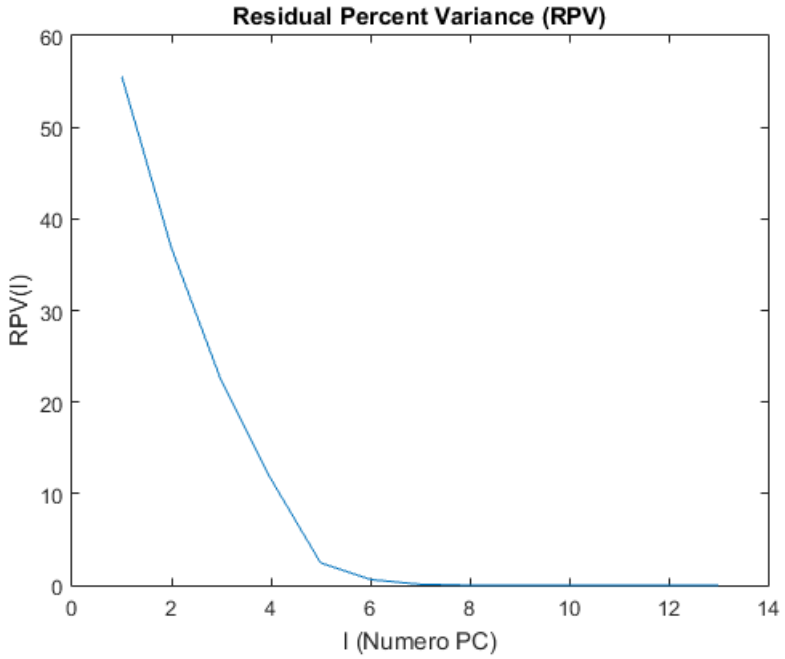


Figura 4.6: Residual Percent Variance con vettore di stato aumentato.

l	$CPV(\%)$
1	44.4
2	63.2
3	77.7
4	88.4
5	97.6
6	99.3
7	99.5
8	99.7
9	99.8
10	99.9
11	99.9
12	99.9
13	100

Tabella 4.1: Cumulative Percent Variance con vettore di stato aumentato.

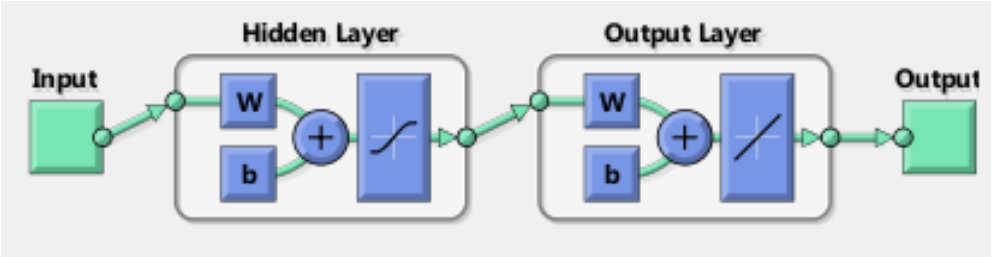


Figura 4.7: Architettura di una rete generica per fitting di funzioni ingresso-uscita.

In figura 4.8 è mostrato l'andamento dell'errore quadratico medio (*mse*) durante l'addestramento della rete neurale approssimante la funzione $G(\cdot)$, mentre in figura 4.9 è riportato l'istogramma dell'errore della stessa rete al termine dell'addestramento.

Possiamo osservare che l'errore quadratico medio finale risulta estremamente basso (dell'ordine di 10^{-6}) e anche l'istogramma dell'errore mostra che la differenza tra l'uscita stimata dalla rete e il target effettivo risulta sempre molto piccola ($< 10^{-2}$).
Di maggiore entità è invece l'errore commesso durante la fase di ricostruzione. La rete che mappa la matrice di *score* T nella matrice dei dati X presenta infatti un mse finale di circa 0.1, come mostrato in figura 4.10.
Il rislutato può comunque essere considerato molto buono dato l'elevato *r-square* del fitting output-target, che si mantiene sempre maggiore di 0.96 (figura 4.11).

Prima di procedere con la presentazione dei risultati ottenuti, è opportuno ribadire un concetto estremamente importante: quando parliamo del vettore dei dati x, supponiamo sempre che questo sia a media nulla e a varianza unitaria. Durante l'elaborazione *off-line* del dataset X, è necessario perciò memorizzare la media e la varianza dei dati per poter successivamente "normalizzare" le variabili misurate *on-line*. Questo aspetto assume particolare rilevanza quando passiamo dal siste-

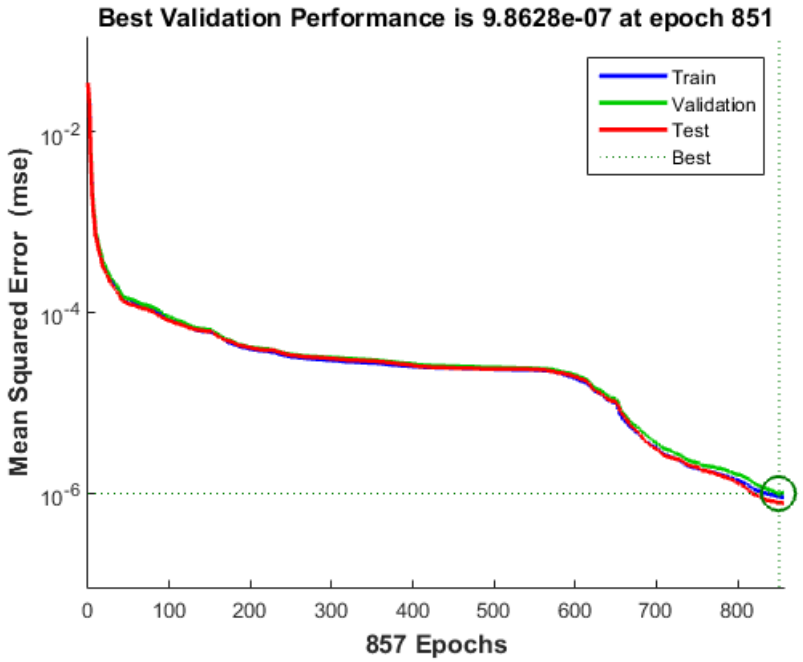


Figura 4.8: Errore quadratico medio durante l'addestramento della rete neurale che approssima la funzione $G(\cdot)$.

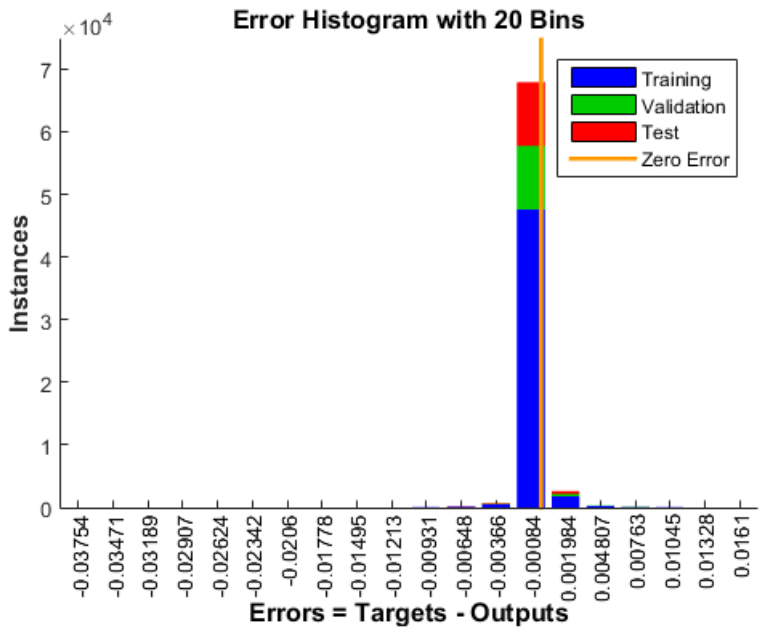


Figura 4.9: Istogramma dell'errore della rete neurale che approssima la funzione $G(\cdot)$.

ma ideale al sistema fisico, nel quale le misure acquisite dai sensori sono necessariamente affette da rumore. Il rumore infatti potrebbe introdurre dei cambiamenti nel valor medio (bias) o nella varianza dei dati misurati, che se non vengono presi in considerazione possono portare alla generazione di falsi allarmi o al mancato riconoscimento di alcuni guasti durante l'applicazione degli algoritmi di FD e FI.
A questo punto non ci resta che mostrare i risultanti ottenuti simulando il comportamento del

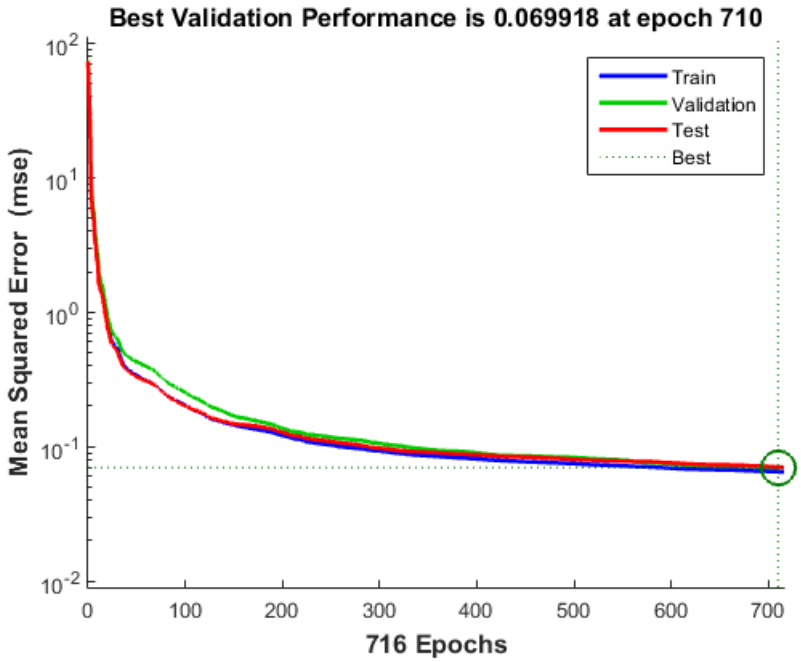


Figura 4.10: Errore quadratico medio durante l'addestramento della rete neurale che approssima la funzione $F(\cdot)$.

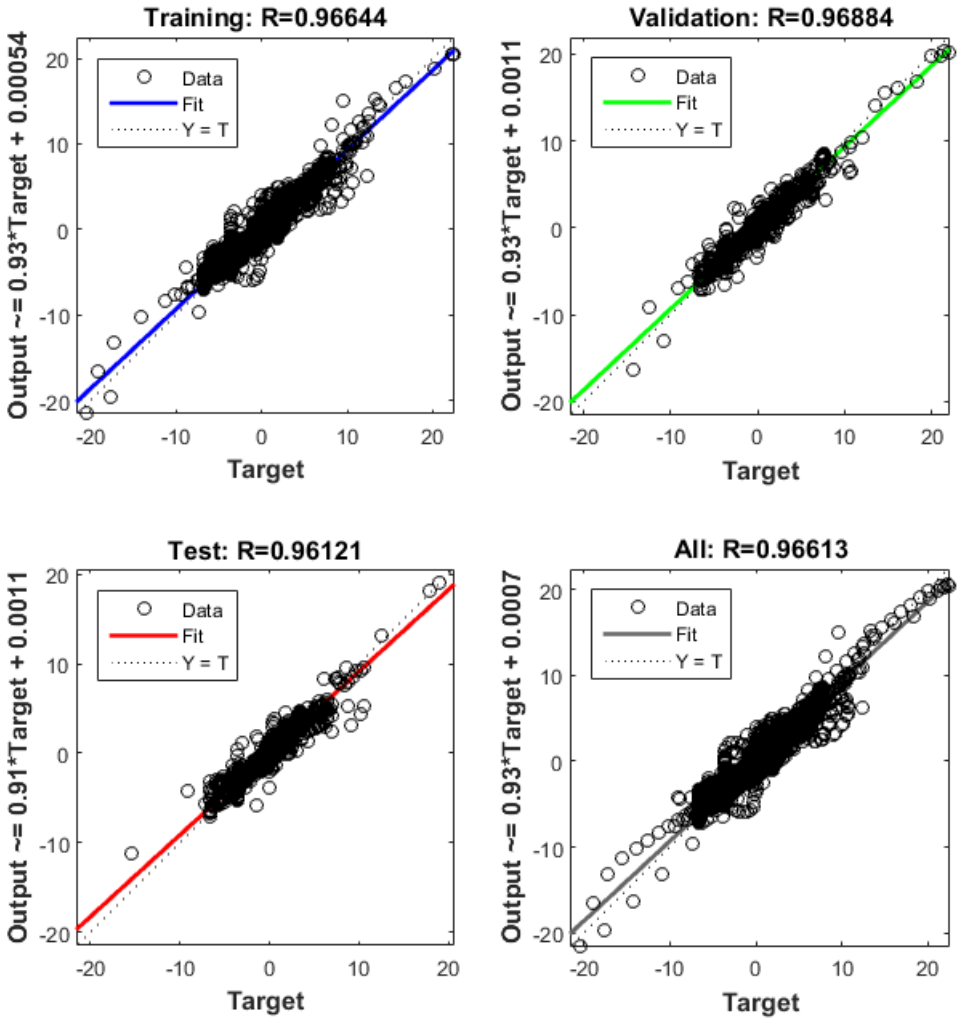
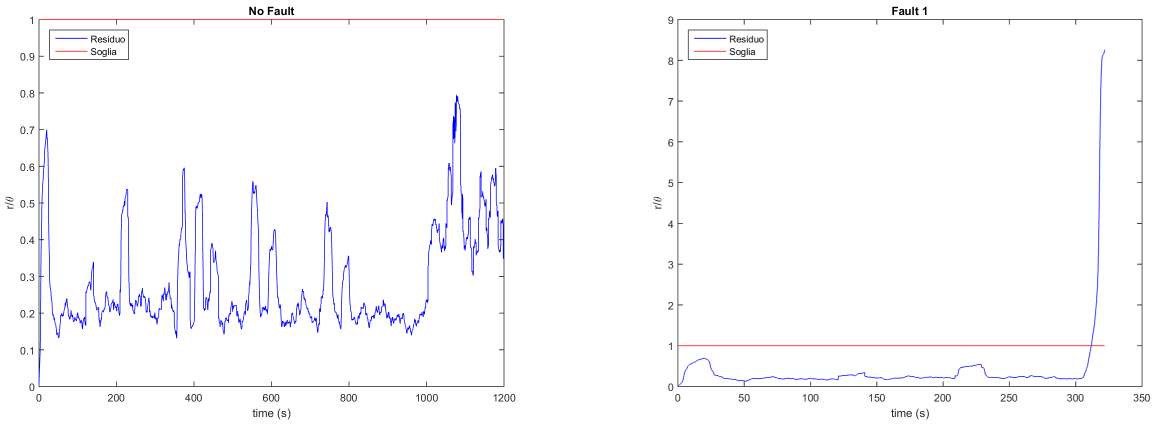
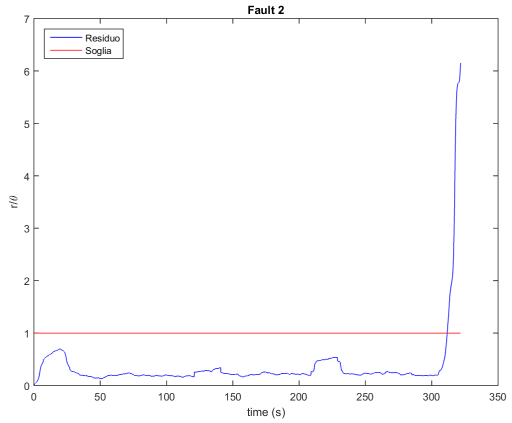


Figura 4.11: Fitting target-output per la rete neurale che approssima la funzione $F(\cdot)$.

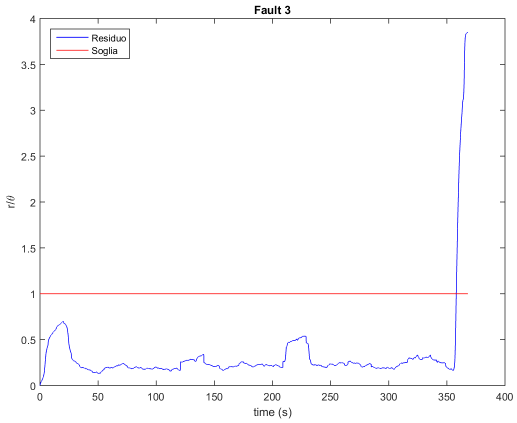
veicolo durante la survey descritta nel paragrafo 3.1 e utilizzando la tecnica della NLPCA.



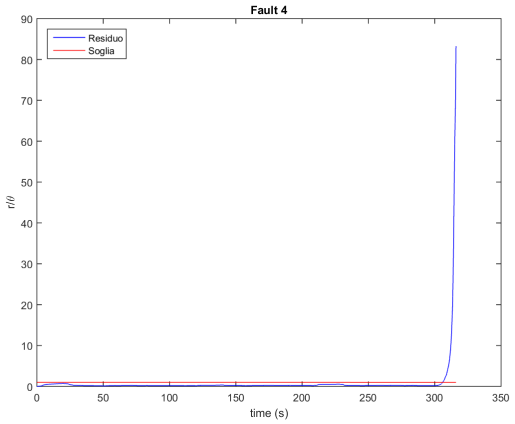
(a) *No fault*. (b) *Fault 1*.



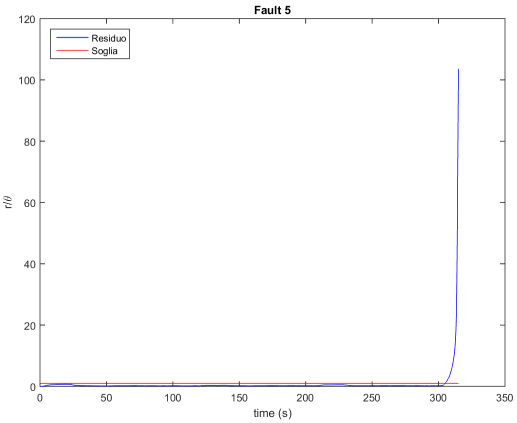
(c) *Fault 2*.



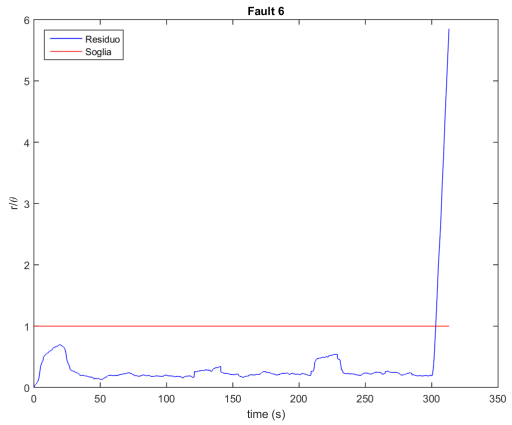
(d) *Fault 3*.



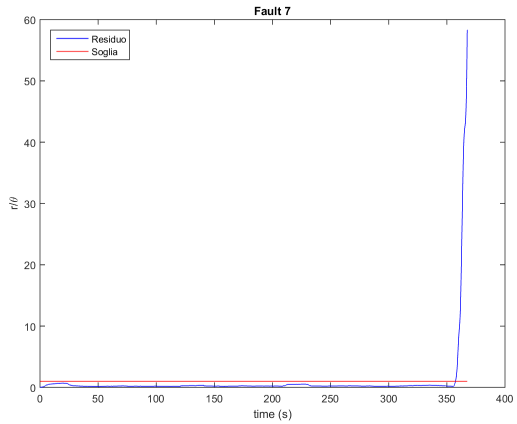
(e) *Fault 4*.



(f) *Fault 5*.



(g) *Fault 6*.



(h) *Fault 7*.

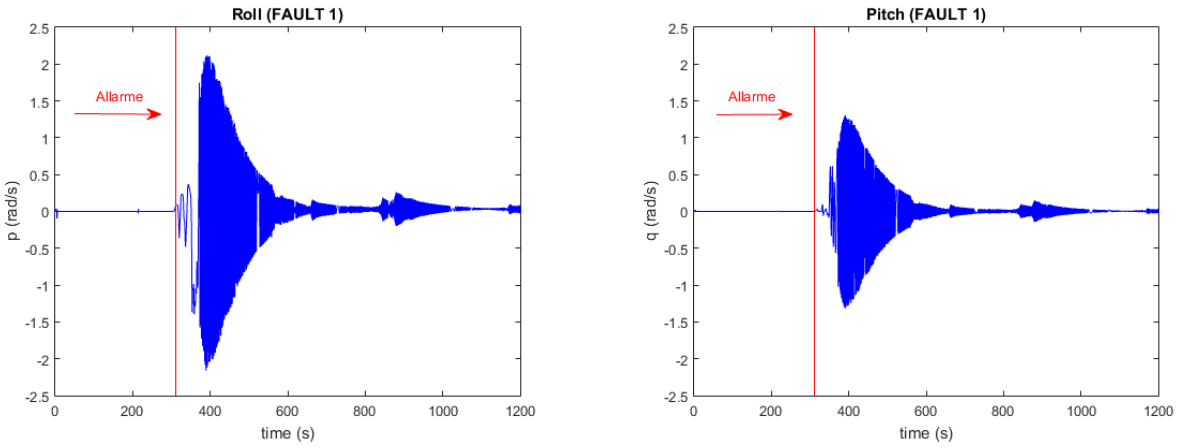
Figura 4.12: r/θ al variare del guasto. Il fault è stato simulato a partire dall'istante $t = 300$ s, mentre la simulazione è stata interrota dopo 10 s dalla rilevazione del guasto.

In figura 4.12 è mostrato l'andamento nel tempo del residuo r, scalato per un'opportuna soglia θ, al variare della situazione in cui si trova il veicolo.
Come già descritto nel paragrafo 4.3, il residuo altro non è che il modulo al quadrato della differenza tra vettore misurato x e vettore ricostruito $\hat{x}$ tramite l'utilizzo della mappa autoassociativa in figura 4.5, costituita dalle due reti neurali precdenti poste in cascata. Possiamo osservare che

quando nessun guasto è presente (figura 4.12a), il residuo rimane sotto la soglia d'allarme (linea rossa) per l'intera durata della survey. Gli altri grafici di figura 4.12, sono stati ottenuti simulando un guasto su ognuno dei thruster all'istante $t = 300\ [s]$. Si può notare come il guasto venga rilevato dopo pochi secondi dalla rottura dell'attuatore. Fanno chiaramente eccezione i thruster 3 e 7, la cui rilevazione avviene solo quando il veicolo deve curvare, dal momento che è l'unica occasione in cui tali attuatori sono utilizzati.
Come per l'algoritmo di FD tramite PCA, anche in questo caso il guasto è sempre rilevato prima che il veicolo diventi instabile. A titolo di esempio si possono osservare i grafici in figura 4.13 nei quali è mostrato l'andamento delle velocità di *roll* e *pitch* in presenza di un guasto sul primo attuatore (simulato a partire dall'istante $t = 300\ [s]$) e l'istante nel quale il segnale di allarme è stato generato.

4.6 Fault Isolation con NLPCA

Nel paragrafo precedente abbiamo visto come la tecnica della NLPCA permetta di risolvere il problema di FD sui thruster di un veicolo subacqueo sovrattuato come V-Fides. Resta ancora da capire come utilizzare tale strumento per individuare il guasto effettivamente avvenuto nel sistema. In letteratura esistono vari metodi per effetuare FI con NLPCA, un esempio particolarmente interessante è la *Partial NLPCA*, ovvero una variante non lineare della *Partial PCA* descritta in [2, 9]. Tutte le metodologie analizzate, però, hanno un aspetto comune, ovvero prevedono che ogni guasto influenzi solo alcune delle variabili del sistema, tolte le quali il vettore dei dati mantiene la struttura di correlazione presentata durante le condizioni di funzionamento normale. Se ciò può andar bene per processi industriali fortemente disaccoppiati, per il nostro sistema in cui, grazie alla presenza degli anelli di controllo, ogni guasto va ad influire su tutte le variabili, questo aspetto rappresenta davvero un grosso limite. Tale problema è lo stesso che è già stato incontrato con la PCA e come nel caso precedente abbiamo a disposizione due possibili soluzioni. La prima è quella di utilizzare l'algoritmo della NLPCA solo per rilevare la presenza di un guasto, mentre l'isolamento sarà fatto con una tecnica differente, come ad esempio un classificatore neurale analogo a quello presentato nel paragrafo 3.5.2 .



(a) *Velocità di rollio (p).* (b) *Velocità di beccheggio (q).*

Figura 4.13: Velocità di roll e pitch (blu) in presenza di un guasto sul primo attuatore (simulato a partire dall'istante $t = 300\ s$) e istante nel quale il segnale d'allarme (rosso) è stato generato.

Un'altra possibilità più interessante è quella di sfruttare la conoscenza sul modello dinamico del veicolo, per prevederne il comportamento in presenza di guasti. In pratica l'idea è quella di simulare la rottura di un generico attuatore i di V-Fides e memorizzare le variabili di nostro interesse per costruire un nuovo dataset X_i. Il generico vettore dei dati $x_i \in \mathbb{R}^{12}$ può essere scritto come

$$x_i = \begin{bmatrix} u_{cmd_i} \\ \nu \end{bmatrix} \tag{4.13}$$

dove $u_{cmd_i} \in \mathbb{R}^6$ è il vettore degli ingressi agli attuatori in cui la componente i è stata rimossa.
A questo punto si può procedere con il calcolo *off-line* delle curve principali corrispondenti e quindi con la determinazione della matrice di score T_i. Disponendo di entrambe le matrici di dati, possiamo costruire le due reti neurali in figura 4.3 che approssimano rispettivamente le funzioni $G_i(\cdot)$ e $F_i(\cdot)$ tali che

$$T_i = G_i(X_i) \tag{4.14}$$

$$\hat{X}_i = F_i(T_i) \ . \tag{4.15}$$

Ponendo in sequenza le due reti neurali, otteniamo una mappa autoassociativa che rappresenta un modello NLPCA del veicolo in presenza di un guasto sull'i-esimo thruster. Dato un ingresso x_i alla mappa, questa restituisce in uscita un vettore ricostruito $\hat{x}_i$, che "assomiglierà" al vettore di partenza se e solo se l'i-esimo attuatore del veicolo è effettivamente rotto. Definendo quindi il residuo

$$R_i = \|x_i - \hat{x}_i\|^2 \ , \tag{4.16}$$

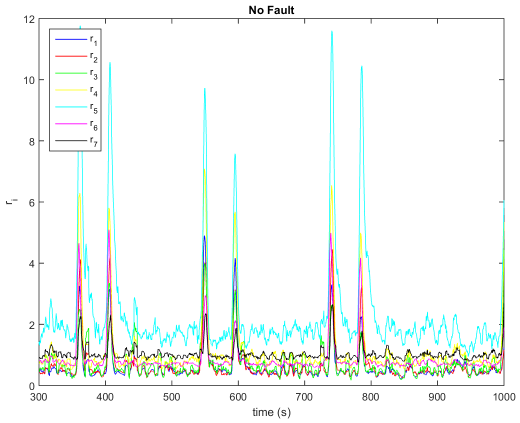
avremo che se il modulo di FD rileva la presenza di un guasto e se $R_i < \theta_i$, dove θ_i è una soglia opportuna, allora possiamo concludere che il thruster rotto è proprio quello associato al residuo.
Ripetendo questa procedura per ogni guasto, otteniamo un set costituito da 8 differenti modelli NLPCA (un modello per la FD e 7 per la FI). Tale strategia è concettualmente identica a quella presentata nel paragrafo 3.5.1, con l'unica differenza che si è deciso di modellare il comportamento del veicolo in presenza di un guasto con una tecnica non lineare.
Dal momento che alcuni guasti hanno effetti simili sul sistema, è possibile che quando il modulo di FD rilevi la presenza di un thruster rotto, ci siano più residui R_i al di sotto della rispettiva soglia. Questo produce un'ambiguità, non permettendo di decidere quale sia l'attuatore effettivamente danneggiato. Per risolvere questo problema si è deciso di utilizzare dei residui "normalizzati" definiti come

$$r_i = \frac{R_i}{\theta_i} \ , \tag{4.17}$$

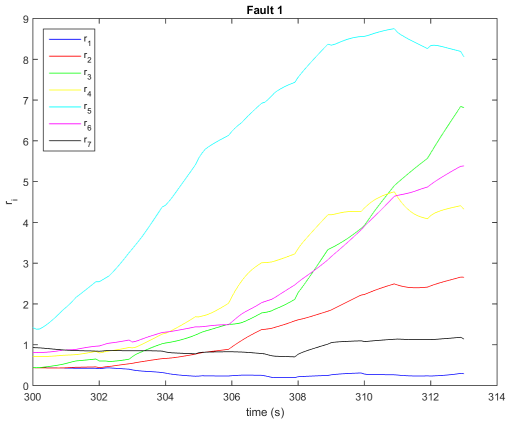
dove θ_i è la soglia associata al residuo R_i e determinata in base al suo valore in condizioni nominali. Questo serve per rendere tutti i residui confrontabili tra di loro. Per individuare il guasto sul veicolo, basterà semplicemente osservare qual è il residuo normalizzato r_i che presenta il valore più piccolo al momento della rilevazione del fault.
In figura 4.14 è mostrato il comportamento dei residui normalizzati durante la simulazione della survey. Possiamo osservare che se nessun guasto è presente (figura 4.14a) allora i residui presentano dei valori molto vicini senza quindi una marcata differenza tra di loro. Negli altri grafici invece sono riportati gli andamenti dei residui in seguito al verificarsi della rottura di un attuatore (simulata a partire dall'istante $t = 300\ [s]$). È evidente che il residuo corrispondente al thruster danneggiato subisce una rapida decrescita, mentre gli altri tendono ad aumentare. Fanno eccezione, come facilmente prevedibile, i thruster 3 e 7 che diventano quelli con valore minore solo superati

i 350 $[s]$. Questo in realtà non costituisce un grande problema dal momento che tali attuatori sono utilizzati solo quando il veicolo deve curvare e quindi, come mostrato nel grafico in figura 4.12, la loro rottura viene rilevata in ritardo rispetto a quella degli altri.
Possiamo perciò osservare come una rappresentazione del sistema mediante curve principali, permetta di risolvere correttamente sia il problema di FD che di FI per tutti i thruster del veicolo V-Fides.

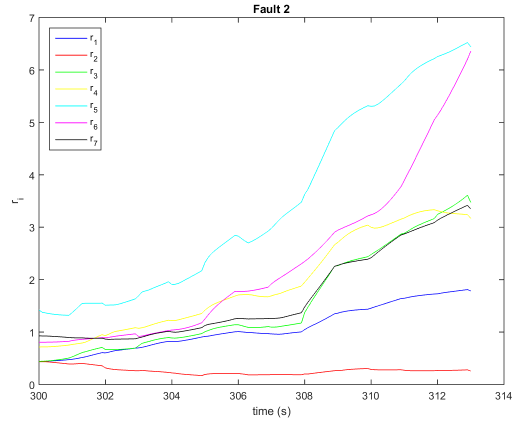
Nel paragrafo precedente abbiamo sottolineato quanto sia importante normalizzare il vettore dei dati x per poter applicare l'algoritmo di FD. Tale aspetto diventa ancora più rilevante nel caso della FI. Quest'ultima, infatti, a differenza della precedente, è fortemente dipendente dal modello del sistema, dal momento che il dataset X_i può essere acquisito solo attraverso delle simulazioni. Se vogliamo normalizzare correttamente i vettori $x_i(t)$ misurati *on-line* durante la missione, non basta conoscere la media e la varianza del dataset teorico X_i calcolato in fase di simulazione, ma sarà necessario anche stimare le caratteristiche del rumore suille variabili di nostro interesse.



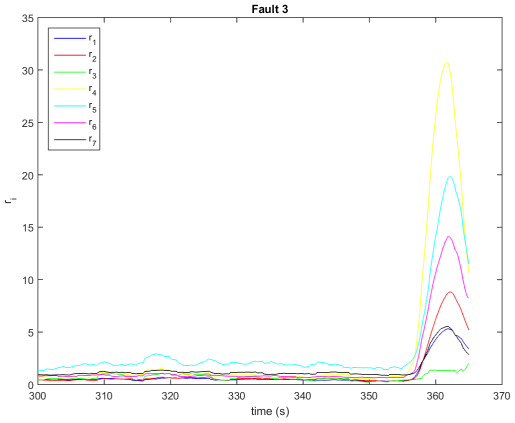
(a) *No fault*.



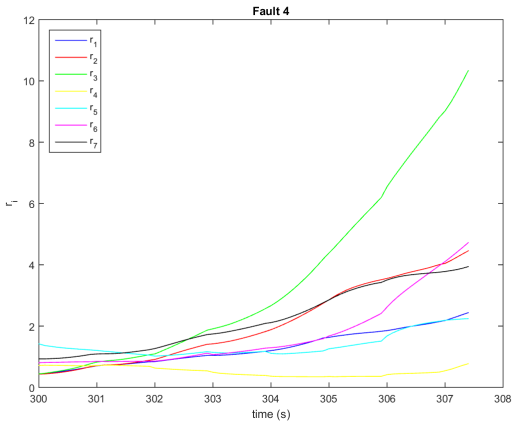
(b) *Fault 1*.



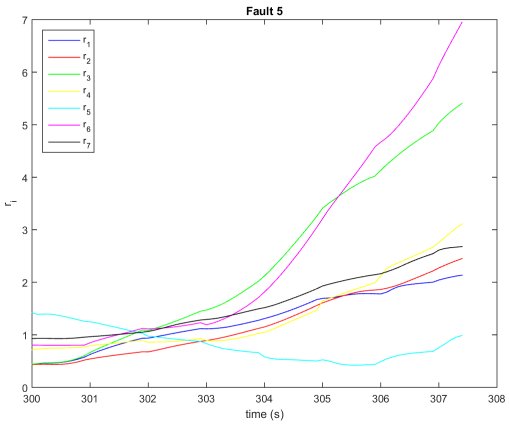
(c) *Fault 2*.



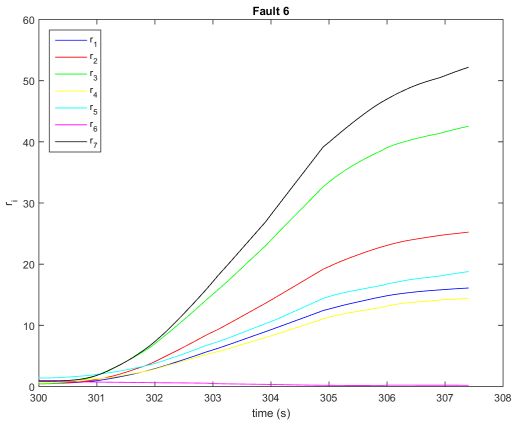
(d) *Fault 3*.



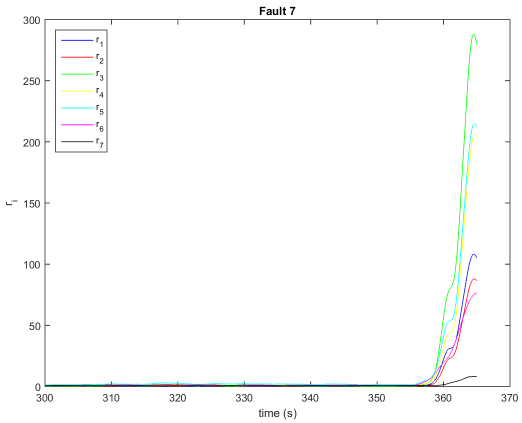
(e) *Fault 4*.



(f) *Fault 5*.



(g) *Fault 6*.



(h) *Fault 7*.

Figura 4.14: r_i al variare del guasto. Il fault è stato simulato a partire dall'istante $t = 300\ s$.

CAPITOLO 5

CONFRONTO ALGORITMI

Nei capitoli 3 e 4 sono stati mostrati i risultati ottenuti applicando due differenti tecniche di FDI per la rilevazione e l'isolamento dei guasti sui thruster di un veicolo subacqueo sovrattuato autonomo. Tali tecniche sono state applicate supponendo sia una conoscenza perfetta del modello del sistema, sia che la traiettoria desiderata per il veicolo si mantenga sempre la stessa.
In questo capitolo cercheremo invece di mettere a confronto le prestazioni dei due metodi, soffermandoci in particolare sulla loro velocità di risposta nell'individuazione di un eventuale guasto e sulle loro capacità di generalizzare (ovvero di riconoscere anomalie anche in traiettorie differenti rispetto a quello utilizzata durante l'addestramento). Si valuterà, inoltre, la robustezza dei due algoritmi proposti ai cambiamenti nei parametri del modello.

5.1 Fault Recovery

Uno dei motivi principali per il quale è importante disporre di un modulo di FDI su un veicolo autonomo sovrattuato è che il riconoscimento precoce di un eventuale guasto su un attuatore può permettere al sistema di controllo di modificare la sua azione in modo da compensare la rottura del thruster. In particolare avremo che se ad un dato istante viene individuata la presenza di un guasto sul thruster i, allora il modulo di fault recovery si occuperà di informare il sistema di allocazione del controllo (descritto nel paragrafo 2.2) il quale calcolerà il vettore dei comandi u_{cmd} escludendo dal processo di ottimizzazione la componente i-esima di u_{cmd} che verrà forzata a 0.
Dal momento che il sistema di fault recovery è lo stesso sia nel caso di FDI tramite PCA sia tramite NLPCA, avrà prestazioni migliori l'algoritmo che è in grado di riconoscere il guasto più velocemente, ovvero quello che riuscirà ad attivare per primo la funzione di recupero.
In tabella 5.1 sono mostrati i tempi trascorsi tra la rottura di un attuatore (simulata a partire dall'istante $t = 300\ [s]$) e il suo riconoscimento con ognuno dei due metodi.

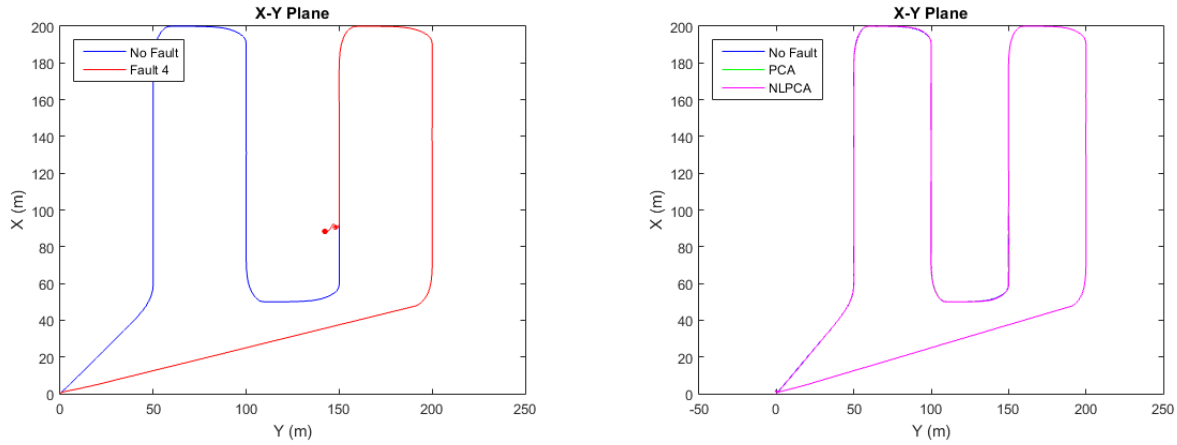
Dai risultati in tabella è evidente che la tecnica della PCA permette di rilevare il guasto quasi sempre più rapidamente rispetto alla sua variante non lineare, anche se con uno scarto molto pic-

	Δt_{PCA}	Δt_{NLPCA}	Best Δt
Thruster 1	9.1	11.7	*PCA*
Thruster 2	10.6	11.7	*PCA*
Thruster 3	62.5	58.1	*NLPCA*
Thruster 4	3.6	6.0	*PCA*
Thruster 5	3.6	5.0	*PCA*
Thruster 6	9.6	3.0	*NLPCA*
Thruster 7	57.0	57.5	*PCA*

Tabella 5.1: Nella prima colonna è riportato il thruster danneggiato (fault simulato all'istante $t = 300\ s$); nelle colonne due e tre è mostrato il tempo impiegato per la rilevazione del guasto utilizzando rispettivamente la tecnica della PCA e della NLPCA; nell'ultima colonna è mostrato, infine, quale delle due tecniche individua il guasto nel tempo minore.

colo. Solo la rottura degli attuatori 3 e 6 viene individuata più rapidamente tramite la NLPCA. Nel paragrafo 3.2.2 è stato mostrato però, che il veicolo è in grado di completare la survey con delle buone prestazioni anche con il terzo thruster rotto e senza che sia presente alcun sistema di fault recovery. D'altro canto, nel paragrafo 3.2.4, è stato già messo in luce come la rottura del sesto motore non possa in alcun modo essere compensata: infatti le forze idrodinamiche tenderanno necessariamente a destabilizzare il veicolo una volta che questi ha iniziato a rallentare in corrispondenza dell'ultimo waypoint.
La velocità di risposta della NLPCA risulta perciò maggiore solo negli unici due casi in cui il sistema di fault recovery non è strettamente necessario al completamento della missione.

Vediamo a questo punto, se questo lieve ritardo nel tempo impiegato per la rilevazione di un guasto comporta effettivamene una significativa riduzione delle prestazione della NLPCA rispetto alla PCA. Per fare ciò, osserviamo cosa succede simulando un guasto sul thruster 4 all'istante $t = 500\ [s]$ sia in assenza di un sistema di fault recovery sia in presenza di fault recovery con PCA e NLPCA.
In figura 5.1a è mostrato come il verificarsi di un guasto di questo tipo impedisca al veicolo completare la survey, facendolo uscire dalla traiettoria desiderata e portandolo a ruotare su sé stesso. Solo grazie all'introduzione di un opportuno sistema di fault recovery (figura 5.1b) il veicolo può proseguire lungo il percorso corretto e raggiungere gli waypoint successivi.
Nessuna differenza significativa è osservata in questo caso tra i due algoritmi di FDI, in quanto le linee rappresentanti le traiettorie descritte sono quasi sovrapposte tra di loro. È interessante notare che non vi è alcuna differenza rilevante nemmeno tra il percorso seguito dal veicolo in assenza di guasti e quelli tracciati in presenza di fault recovery sul quarto attuatore.



(a) *No Fault Recovery*. (b) *Fault Recovery*.

Figura 5.1: Traiettoria percorsa dal veicolo sia in assenza di guasti (blu) sia in presenza di un guasto sul thruster 4 simulato all'istante $t = 500\ s$.

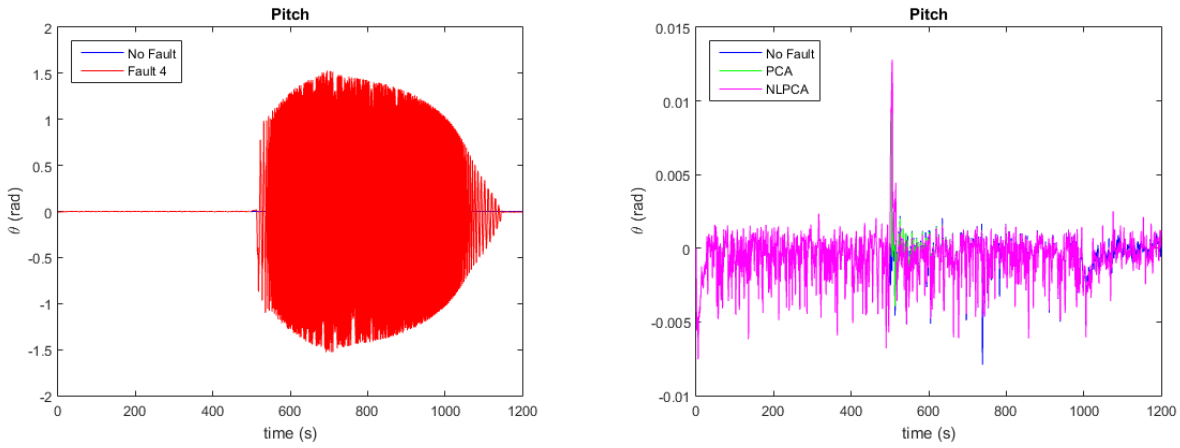
Conclusioni simili sono suggerite anche dai grafici in figura 5.2 nei quali sono riportati gli andamenti nel tempo dell'angolo di beccheggio. Vediamo che anche se con un lieve peggioramento rispetto alla condizione ideale di assenza di guasti, i sistemi di fault recovery riescono comunque a mantenere stabile l'assetto del veicolo.

Un discorso a parte meritano invece i grafici in figura 5.3 che mostrano l'andamento nel tempo dell'angolo di rollio. In figura 5.3a si nota come l'assenza del sistema di fault recovery porta verosimilmente al ribaltamento del veicolo, situazione scongiurata da entrambi i metodi di FDI. Tuttavia, rispetto alla PCA che assicura un angolo di rollio sempre inferiore ai 30° , la NLPCA

presenta un picco nel rollio di poco superiore ai 50°, causato da un ritardo nella determinazione del guasto. Possiamo concludere che la NLPCA pur garantendo la stabilità del veicolo e la riuscita della missione, comporta una maggiore degradazione delle prestazioni rispetto alla PCA.

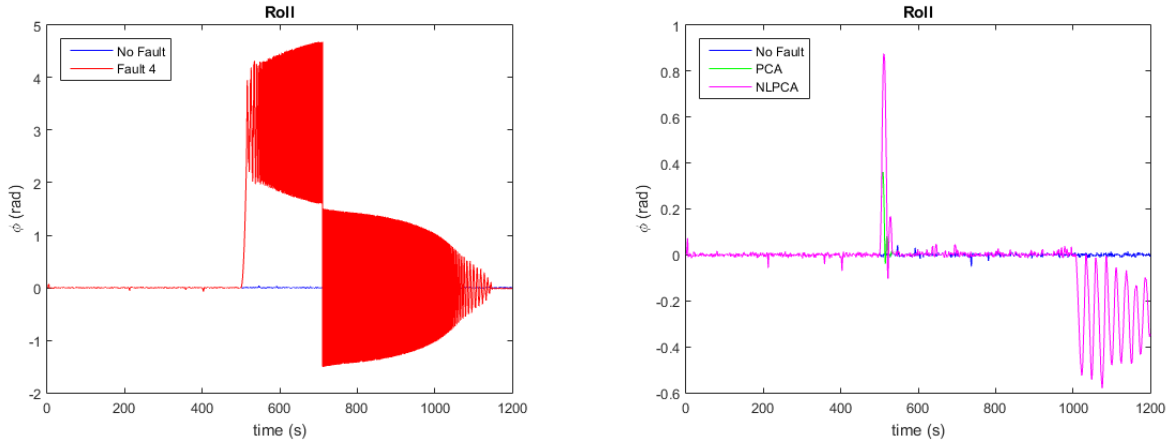
Risultati analoghi possono essere ottenuti simulando la rottura dei thruster 5 e 7. Più problematico è il caso in cui a rompersi sia l'attuatore 1 o 2, dal momento che entrambi i motori risultano fondamentali per il completamento della survey descritta. Come già spiegato nel paragrafo 3.2.1, infatti, i due thruster sono gli unici in grado di esercitare delle forze lungo la direzione di *surge* e la loro mancanza compromette irrimediabilmente le prestazioni del veicolo. La soluzione che abbiamo deciso di adottare prevede che, una volta individuata la presenza di uno dei due guasti, il veicolo venga arrestato nel punto in cui si trova. In questo modo evitiamo che il sistema inizi a ruotare su sé stesso (figura 5.4a) o ad oscillare pericolosamente lungo la direzione di beccheggio (figura 5.5a).

Osservando i grafici riportati, notiamo che anche se il veicolo è costretto a interrompere prematuramente la survey, il sistema di fault recovery ne garantisce comunque la stabilità, a differenza di quanto sarebbe successo se il guasto non fosse stato rilevato in tempo.
In futuro potrebbe essere conveniente studiare una soluzione alternativa, ovvero prevedere un nuovo sistema di guida che si attivi ogni qual volta venga rilevato un guasto sul primo o sul secondo attuatore e che conduca il veicolo alla base ignorando gli waypoint intermedi.



(a) *No Fault Recovery*. (b) *Fault Recovery*.

Figura 5.2: Andamento nel tempo dell'angolo di beccheggio del veicolo sia in assenza di guasti (blu) sia in presenza di un guasto sul thruster 4 simulato all'istante $t = 500\ s$.

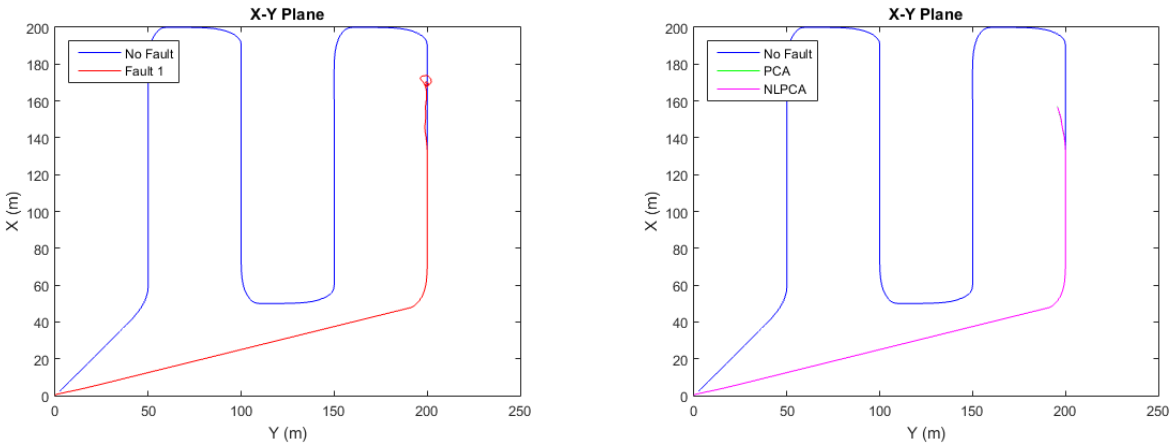


(a) *No Fault Recovery*. (b) *Fault Recovery*.

Figura 5.3: Andamento nel tempo dell'angolo di rollio del veicolo sia in assenza di guasti (blu) sia in presenza di un guasto sul thruster 4 simulato all'istante $t = 500\ s$.

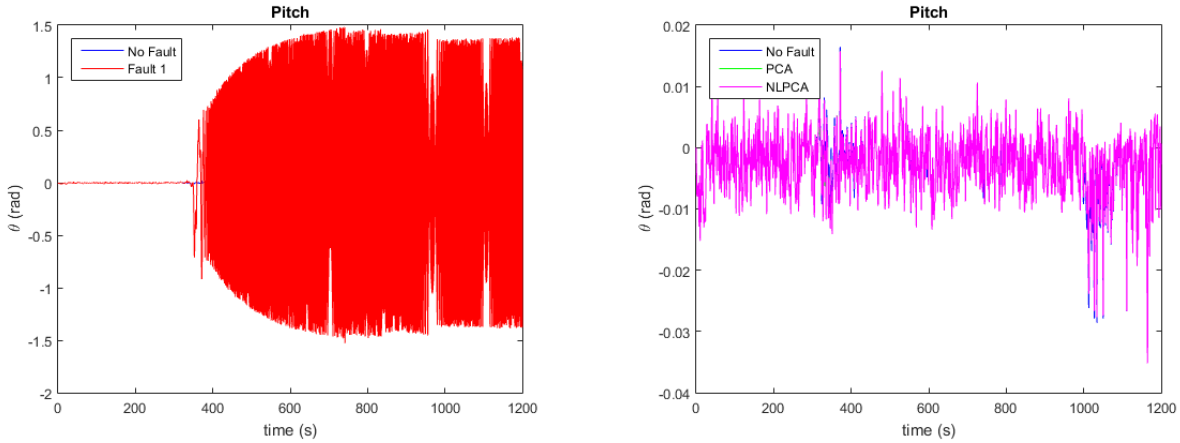
5.2 Cambiamento dei Parametri

Dal momento che tutti i risultati ottenuti sono basati su test effettuati in simulazione, prima di procedere con l'implementazione effettiva dei moduli di FDI sul veicolo è necessario interrogarci sulla robustezza degli algoritmi sviluppati.
Abbiamo già osservato come entrambi gli algoritmi siano in grado di fornire risposte corrette anche in presenza di dati rumorosi: in tutte le simulazioni effettuate, infatti, i dati sono sempre stati contaminati con l'aggiunta di rumore bianco gaussiano al fine di simulare in modo più realistico le misure effettuate dai sensori a bordo del veicolo.
Rimane quindi da vedere cosa succede alla risposta dei moduli di FDI quando questi sono implementati in un sistema "perturbato" rispetto a quello per il quale sono stati addestrati.
Per quanto accurato possa essere, il modello presentato nel capitolo 2 non potrà mai descrivere la



(a) *No Fault Recovery*. (b) *Fault Recovery*.

Figura 5.4: Traiettoria percorsa dal veicolo sia in assenza di guasti (blu) sia in presenza di un guasto sul thruster 1 simulato all'istante $t = 300\ s$.



(a) *No Fault Recovery*. (b) *Fault Recovery*.

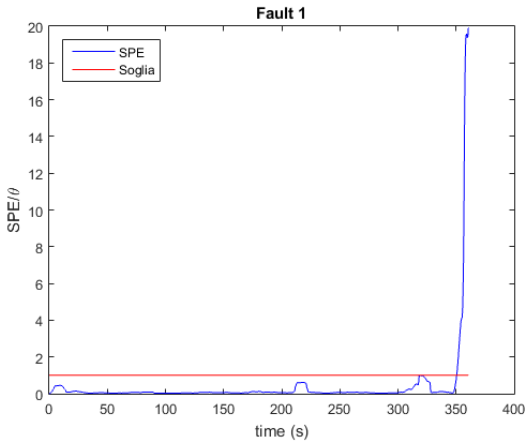
Figura 5.5: Andamento nel tempo dell'angolo di beccheggio del veicolo sia in assenza di guasti (blu) sia in presenza di un guasto sul thruster 1 simulato all'istante $t = 300\ s$.

dinamica di V-Fides in modo perfetto, in quanto ci saranno sempre dei fenomeni non modellati dipendenti ad esempio da fattori ambientali difficilmente controllabili o prevedibili (come correnti marine o la presenza di banchi di pesci). Inoltre anche gli stessi parametri fisici (come le masse, le distanze tra gli attuatori ecc..) non sono noti in modo esatto ma vi è sempre un'incertezza nella conoscenza del loro valore effettivo. Applicando i due moduli di FDI descritti precedentemente ad un modello in cui sono stati variati i parametri dinamici che influenzano maggiormente la dinamica del sistema, abbiamo un'idea di quanto gli algoritmi proposti siano robusti e quanto possa essere affidabile la loro risposta una volta implementati sul sistema fisico reale.
I parametri di maggior interesse per i nostri scopi sono le matrici delle inerzie, delle masse aggiunte e di attrito idrodinamico.
In figura 5.6 sono presentati gli andamenti dei residui SPE (FD con PCA) e le uscite del classificatore neurale (FI) al variare del thruster danneggiato e con una riduzione dei parametri dinamici del 20%. Ogni guasto è stato introdotto all'istante $t = 300\ [s]$ dall'inizio della simulazione.

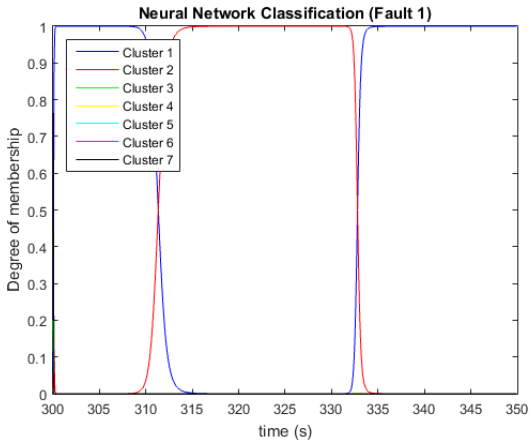
Osservando attentamente i grafici, possiamo notare che non vi è alcuna differenza significativa rispetto al caso nominale per quanto riguarda i thruster dal 3 al 7. Tutti i guasti vengono infatti rilevati e isolati in modo tempestivo, tranne ovviamente il terzo e il settimo attuatore la cui rottura, come già sappiamo, non può essere individuata prima che il veicolo inizi a virare.
Un discorso a parte meritano invece i primi due thruster. I grafici 5.6a e 5.6c mostrano infatti un notevole ritardo nella rilevazione del guasto che in condizioni nominali veniva rilevato circa 10 $[s]$ dopo la sua introduzione nel sistema. In realtà questo non costituisce un grosso problema dal momento che, riducendo i parametri dinamici del modello, il veicolo riesce a mantenere un assetto stabile più a lungo anche senza l'utilizzo del primo o del secondo thruster.

A titolo dimostrativo si possono osservare i grafici in figura 5.7 nei quali sono riportati gli andamenti degli angoli di rollio e beccheggio in presenza di un guasto sul primo attuatore (simulato all'istante $t = 300\ [s]$) fino al momento in cui il segnale d'allarme è generato. È evidente che tali angoli non raggiungono mai valori pericolosi mantenendosi sempre rispettivamente sotto 6° e 1.5°.

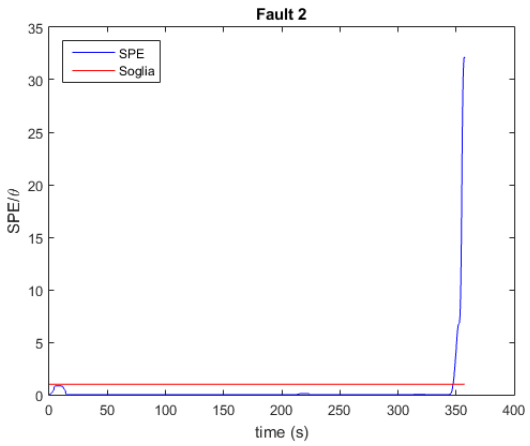
Più delicata è invece la parte riguardante l'solamento dei due guasti. Le figure 5.6b e 5.6d mostrano che, anche se al momento della rilevazione del guasto il classificatore neurale identifica il thruster corretto, vi è un transitorio non trascurabile nel quale il modulo di FI sembra suggerire



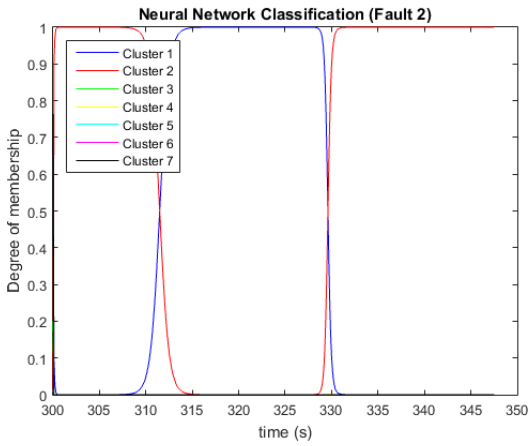
(a) *FD fault 1.*



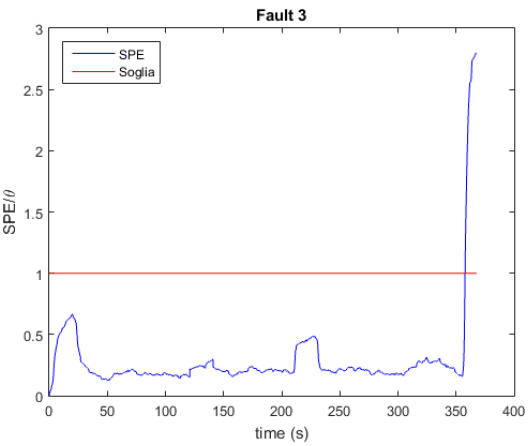
(b) *FI fault 1.*



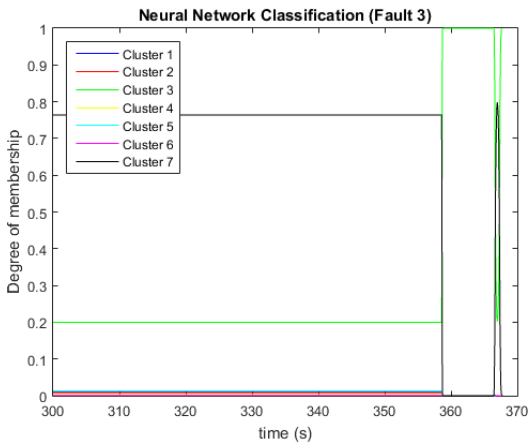
(c) *FD fault 2.*



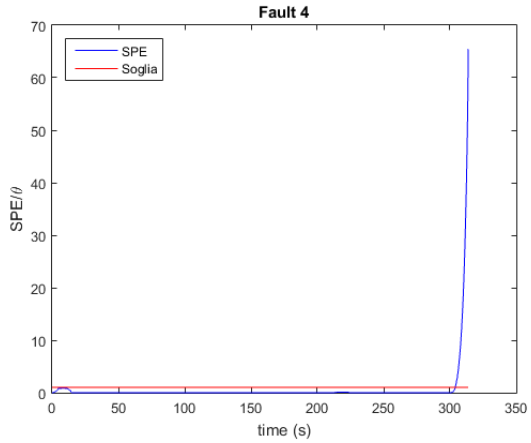
(d) *FI fault 2.*



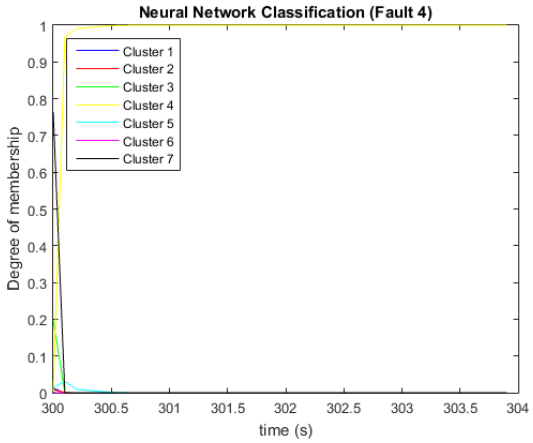
(e) *FD fault 3.*



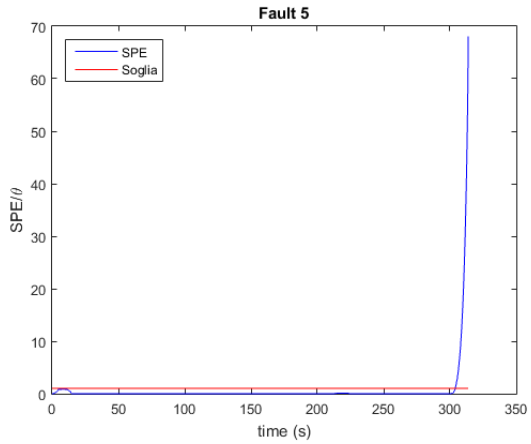
(f) *FI fault 3.*



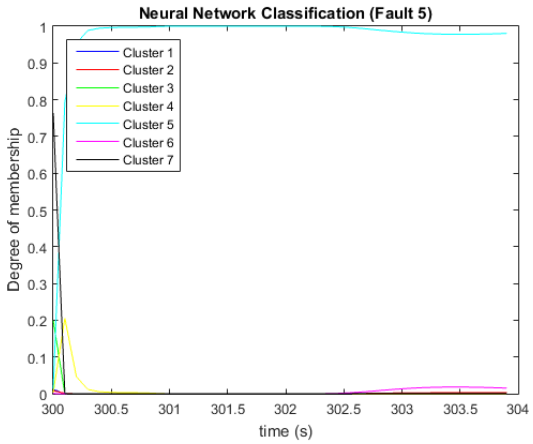
(g) *FD fault 4*.



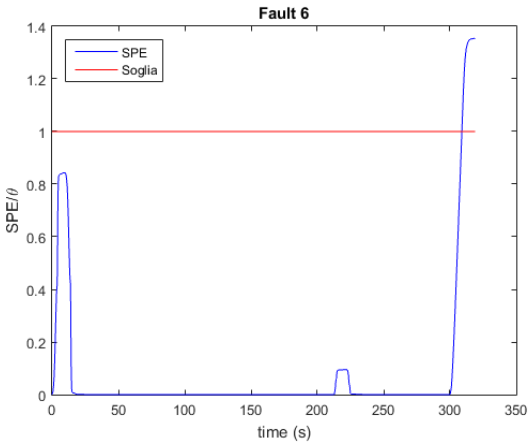
(h) *FI fault 4*.



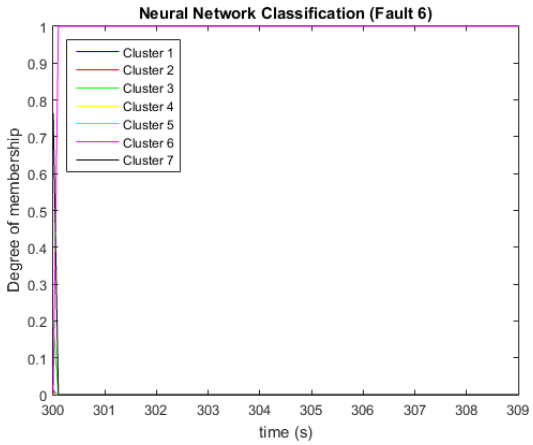
(i) *FD fault 5*.



(j) *FI fault 5*.

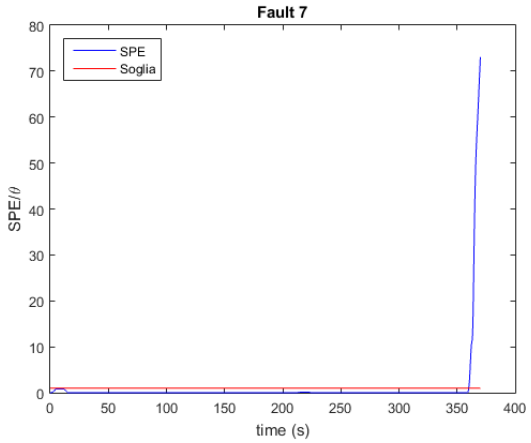


(k) *FD fault 6*.

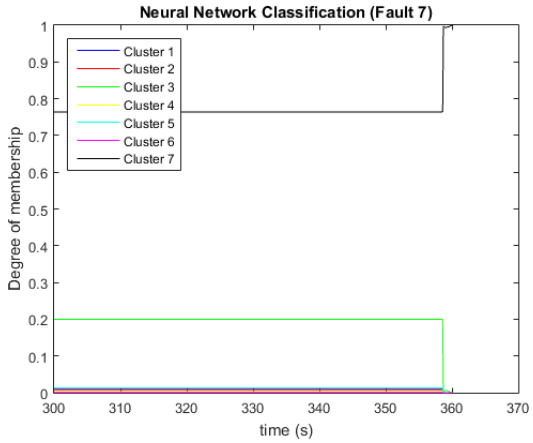


(l) *FI fault 6*.

che si sia rotto l'attuatore sbagliato. In particolare nell'intervallo che va da circa 310 $[s]$ a circa 335 $[s]$ l'algoritmo inverte i due guasti segnalando la rottura del thruster 2 quando a rompersi è stato il thruster 1 e facendo il viceversa con il secondo attuatore.
Vediamo a questo punto cosa succede se, per risolvere il problema di FDI sul sistema perturbato, utilizziamo la tecnica della NLPCA. In figura 5.8 sono presentati gli andamenti dei residui r (FD) e r_i (FI) al variare del thruster danneggiato. Ogni guasto è stato introdotto all'istante $t = 300$ $[s]$ dall'inizio della simulazione.



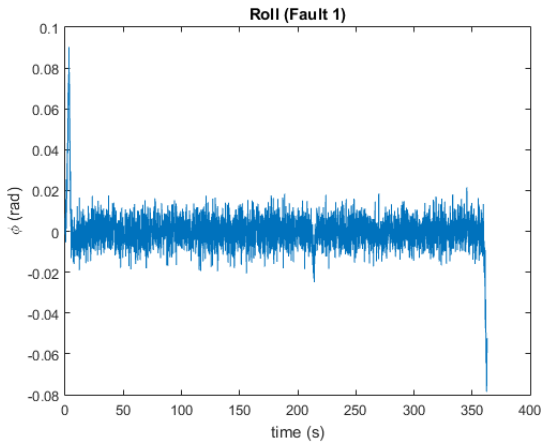
(m) *FD fault 7.*



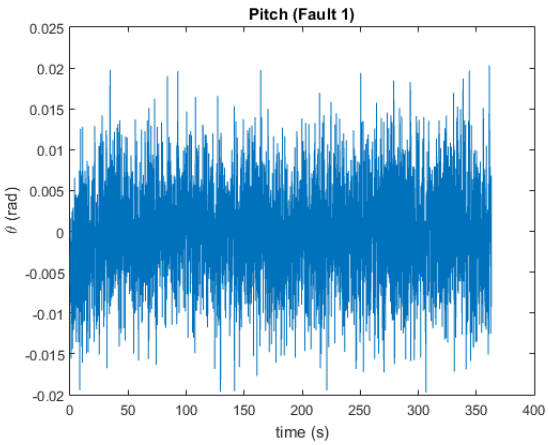
(n) *FI fault 7.*

Figura 5.6: FDI con PCA e classificatore neurale al variare del thruster rotto (guasto simulato a partire dall'istante $t = 300\ s$) con una riduzione del 20% sui parametri dinamici del modello.

Come avvenuto con la PCA, anche utilizzando la NLPCA non si osservano cambiamenti significativi rispetto al caso nominale per quanto riguarda la rilevazione e l'isolamento dei guasti sui thruster dal 3 al 7. Fanno eccezione, nuovamente, solo i thruster 1 e 2 la cui rottura è rilevata con un ritardo notevole (oltre 50 $[s]$). Il ritardo osservato è all'incirca lo stesso di quello presentato dalla PCA e quindi non compromette la stabilità dell'assetto del veicolo.



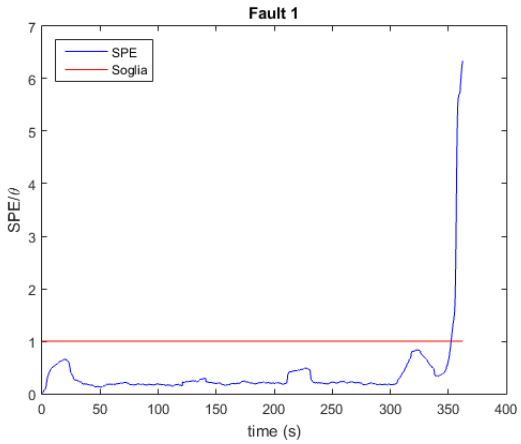
(a) *Roll* (ϕ).



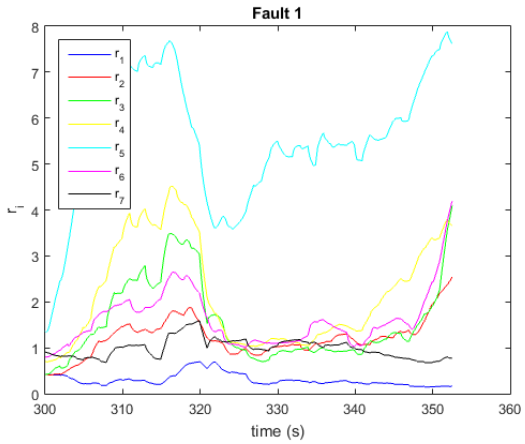
(b) *Pitch* (θ).

Figura 5.7: Andamento nel tempo degli angoli di rollio e beccheggio con una riduzione dei parametri del 20%, in presenza di un guasto sul thruster 1 (simulato a partire da $t = 300\ s$) fino all'istante in cui il guasto è stato rilevato.

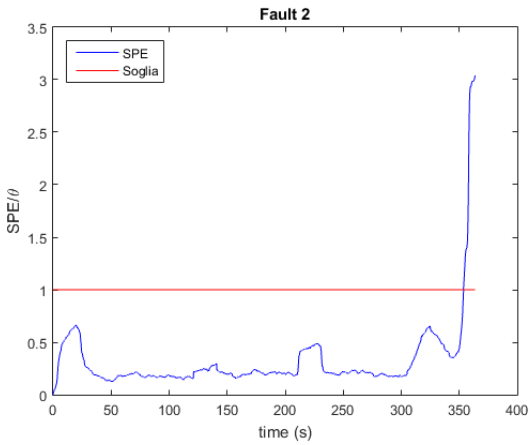
Un significativo miglioramento, rispetto al metodo precedente, si ha per quanto riguarda l'isolamento dei guasti sui primi due motori. Le figure 5.8b e 5.8d mostrano che dopo un brevissimo transitorio, i residui associati ai thruster effettivamente rotti diminuiscono rimanendo quelli a valore minore per tutta l'intervallo temporale che va dall'insorgere del guasto fino alla sua rilevazione. Aumentando i valori dei parametri (anche del 50%) non si osservano differenze apprezzabili rispetto al caso nominale: entrambi gli algoritmi riescono a rilevare e isolare tutti i guasti con i tempi e le modalità presentate nei capitoli precedenti.



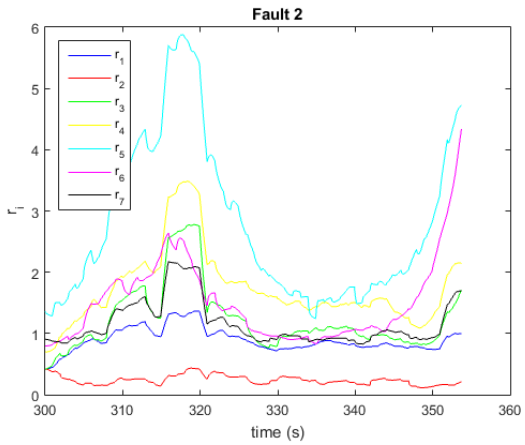
(a) *FD fault 1*.



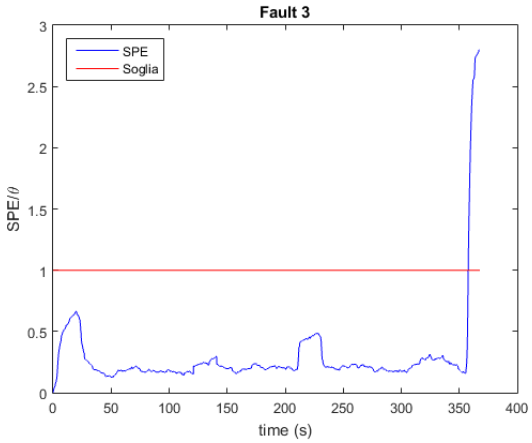
(b) *FI fault 1*.



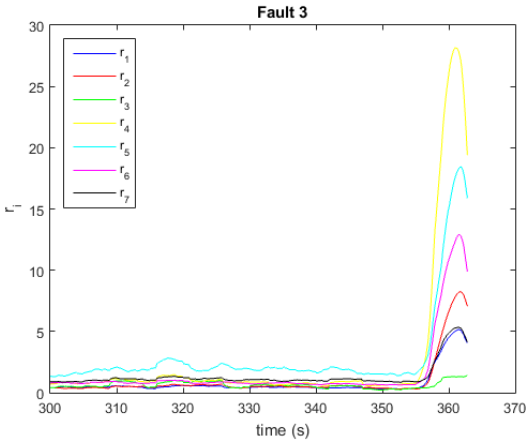
(c) *FD fault 2*.



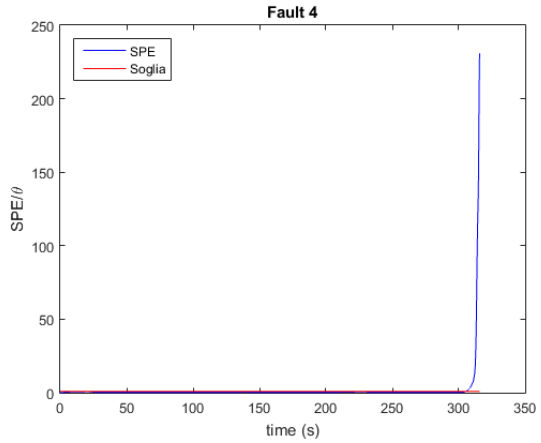
(d) *FI fault 2*.



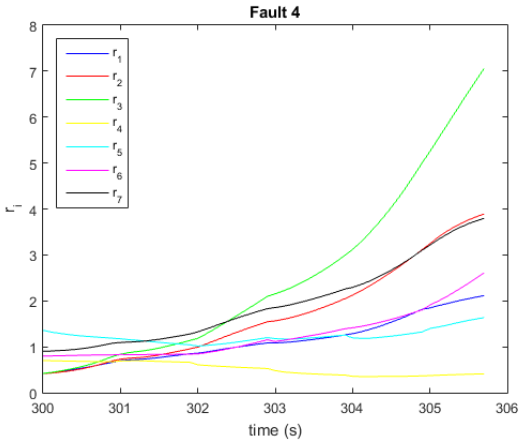
(e) *FD fault 3*.



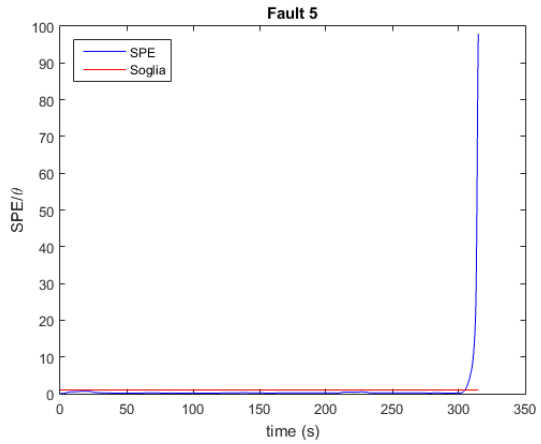
(f) *FI fault 3*.



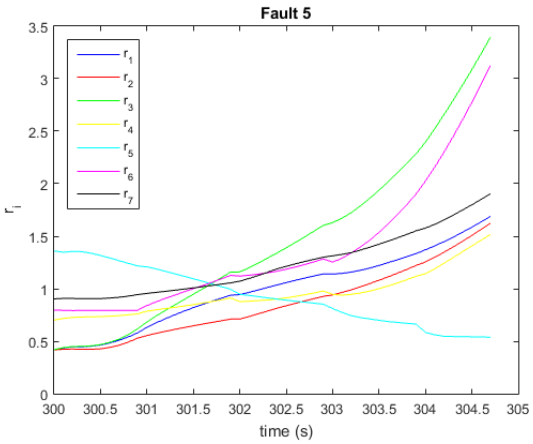
(g) *FD fault 4*.



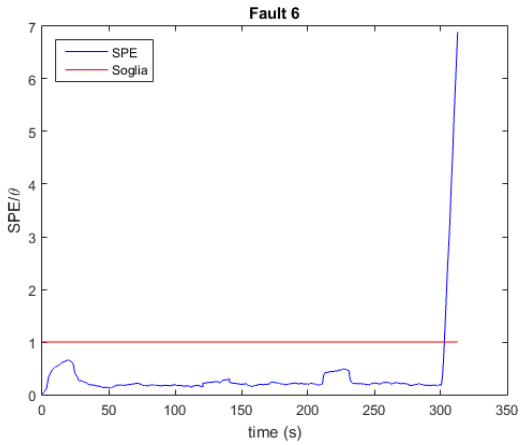
(h) *FI fault 4*.



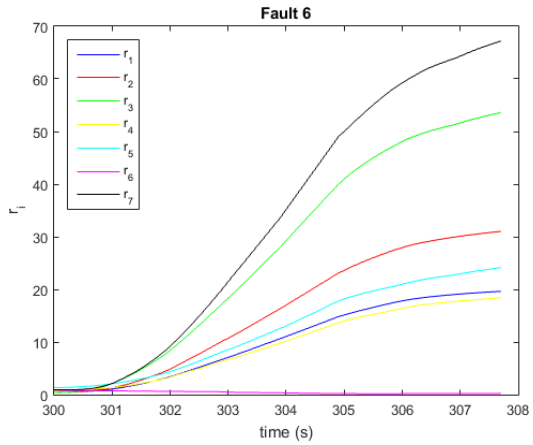
(i) *FD fault 5*.



(j) *FI fault 5*.

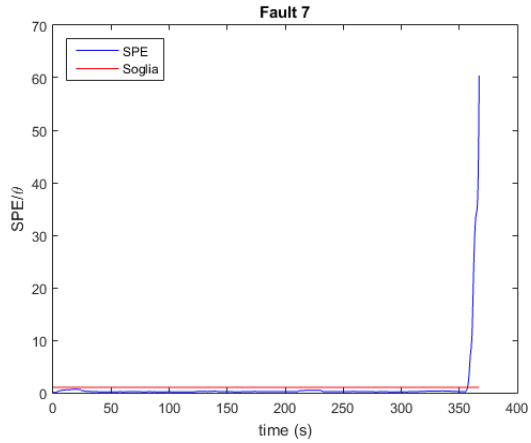


(k) *FD fault 6*.

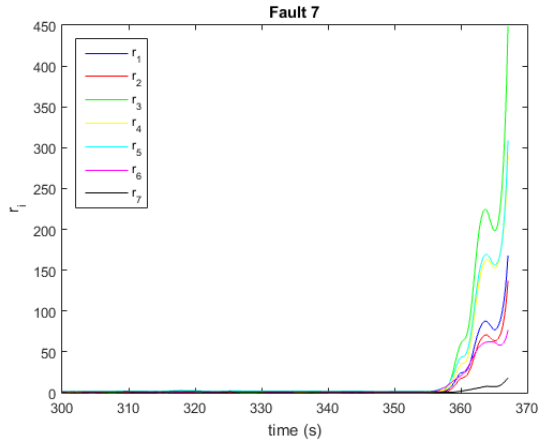


(l) *FI fault 6*.

Da quanto esposto in questo paragrafo possiamo concludere che solo l'algoritmo di FI tramite NLPCA non viene in alcun modo influenzato dalla perturbazione dei parametri di modello, mentre l'uscita del classificatore neurale non sempre può essere ritenuta affidabile (figura 5.6b e 5.6d).
Minore preoccupazione si ha invece per quanto riguarda la robustezza degli algoritmi di FD. Nonostante si sia effettivamente osservato un ritardo notevole nella rilevazione dei guasti sui thruster 1 e 2, abbiamo già spiegato come tale ritardo non vada ad inficiare la stabilità del sistema. Sappiamo, inoltre, che sia la PCA sia la NLPCA sono tecniche model-free e perciò indipendenti dall'accuratezza del modello matematico utilizzato per descrivere il veicolo. Se dovessimo implementare tali



(m) *FD fault 7.*



(n) *FI fault 7.*

Figura 5.8: FDI con NLPCA al variare del thruster rotto (guasto simulato a partire dall'istante $t = 300\ s$) con una riduzione del 20% sui parametri dinamici del modello.

algoritmi di FD sul veicolo reale, il dataset utilizzato per l'addestramento sarebbe, infatti, quello effettivamente acquisito da V-Fides durante un'opportuna survey oceanografica. Perciò piuttosto che interrogarci sulla robustezza degli algoritmi alla perturbazione dei parametri del modello, risulta più opportuno chiederci se il dataset acquisito è sufficientemente esaustivo e permette di modellare tutte le manovre di nostro interesse.
Molto diversa è la situazione riguardo ai moduli di FI. Le tecniche utilizzate in questa trattazione, sono infatti fortemente dipendenti dal modello matematico utilizzato, perciò la robustezza dei due algoritmi è un requisito fondamentale per la loro applicazione.

5.3 Traiettorie Alternative

Nei capitoli precedenti abbiamo già spiegato abbondantemente come le prestazioni di entrambe le tecniche descritte siano fortemente dipendenti dal dataset utilizzato per il loro addestramento. È chiaro, infatti, che manovre in cui la dinamica del veicolo esibisce comportamenti molto diversi da quelli presentati agli algoritmi durante la fase di *training*, saranno riconosciuti come anomalie e quindi genereranno inevitabilmente un segnale di allarme.
L'obiettivo che ci poniamo adesso è quello di verificare se gli algoritmi sviluppati siano o meno in grado di "generalizzare", ovvero di risolvere correttamente il problema di FDI anche su traiettorie differenti, sebbene simili, rispetto a quella utilizzata per l'acquisizione del dataset.

In figura 5.9 è presentata la nuova survey sulla quale andremo a testare l'efficacia dell'algoritmo.

È facile notare come la traiettoria presentata sia strutturalmente identica a quella mostrata in figura 3.1, con l'unica differenza che in questo caso è stato incrementato da due a tre il numero delle "gobbe" e sono state aumentate le lunghezze dei tratti rettilinei.
Questa scelta è stata effettuata sulla base di alcune considerazioni pratiche. Innanzitutto sappiamo che tutte le survey oceanografiche hanno un andamento caratteristico simile tra loro, perciò avrebbe poco senso definire per una survey un percorso molto diverso da quello mostrato. Inoltre è logico supporre che la survey di "prova" utilizzata per l'addestramento dell'algoritmo, abbia una durata minore rispetto alla missione vera e propria. In questo modo, infatti, si evita di perdere troppo tempo e si minimizza la probabilità che il veicolo subisca un guasto prima ancora che su di

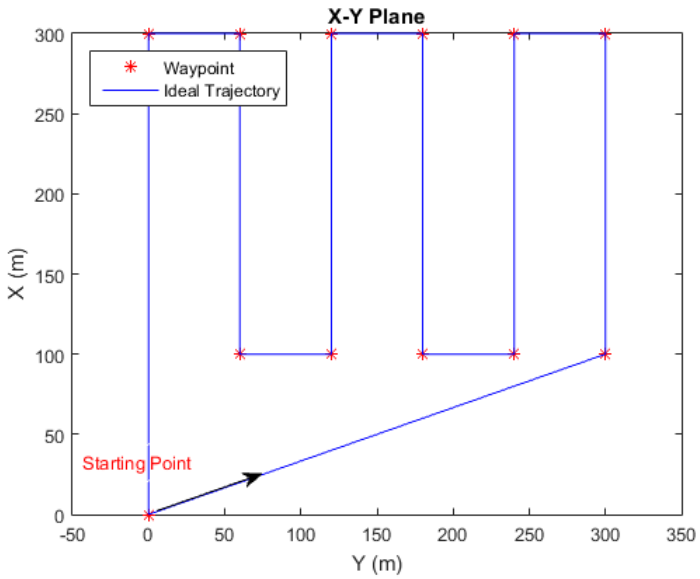


Figura 5.9: Waypoint (rosso) e traiettoria desiderata lungo il piano $X - Y$ (blu) durante la survey utilizzata per testare gli algoritmi di FDI.

esso venga implementato un modulo di FDI.

Osserviamo a questo punto i risultati ottenuti simulando il comportamento del veicolo durante la survey descritta.
Quando tutti gli attuatori funzionano correttamente, entrambi gli algoritmi non rilevano la presenza di alcun guasto. Come evidenziato in figura 5.11, infatti, entrambi i residui si mantengono sotto soglia per tutta la durata della survey.

Vediamo adesso come si comporta il sistema quando andiamo a simulare la rottura dei vari

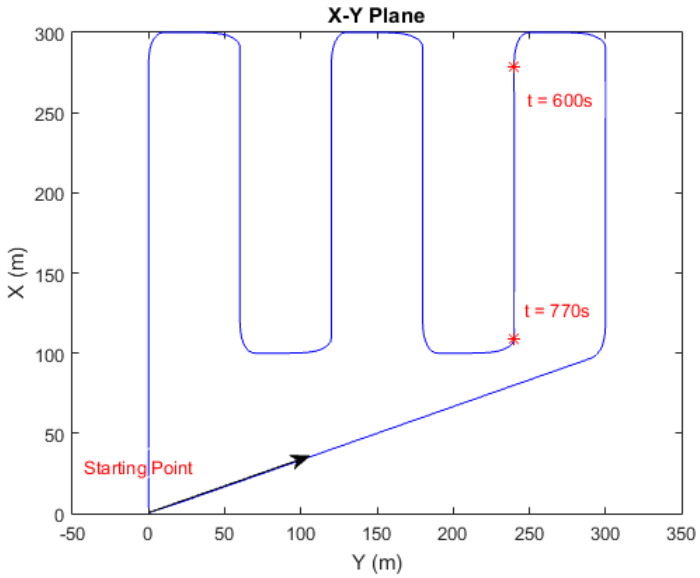
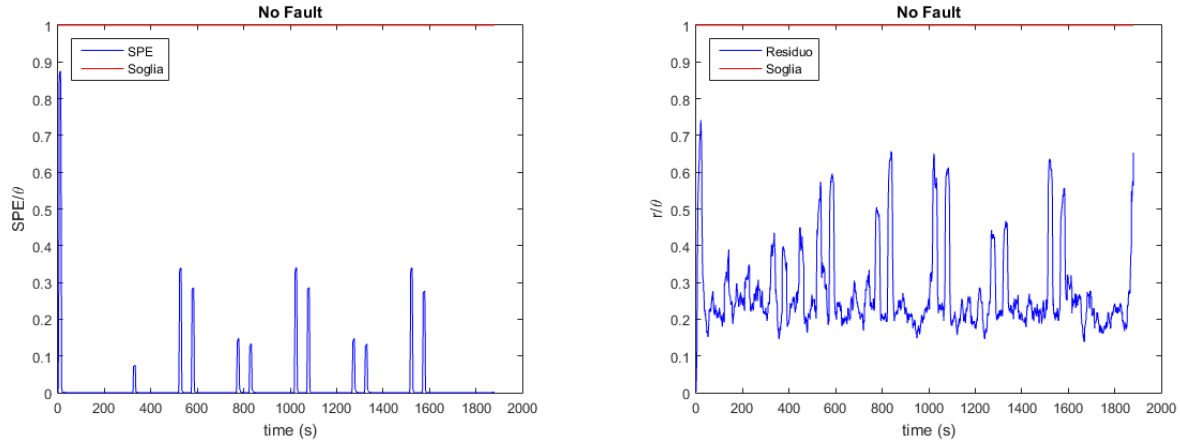


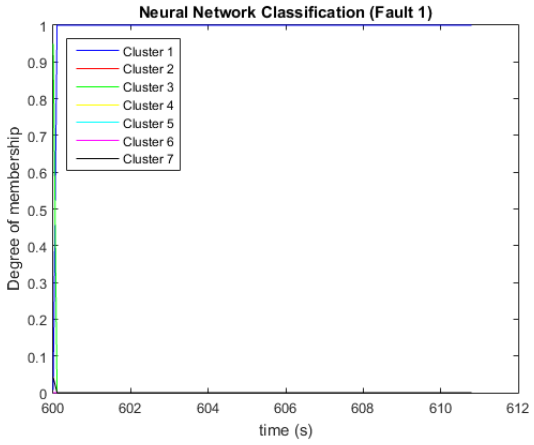
Figura 5.10: Traiettoria percorsa lungo il piano $X - Y$ (blu) in assenza di guasti, durante la survey utilizzata per testare gli algoritmi di FDI.



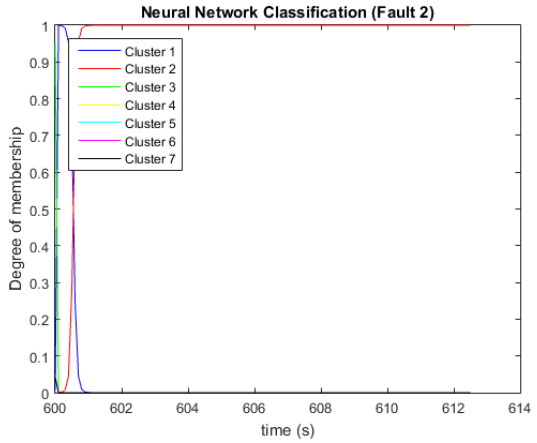
(a) *FD con PCA.* (b) *FD con NLPCA.*

Figura 5.11: Andamento nel tempo dei residui per Fault Detection (sia con PCA sia con NLPCA) (blu) e relativa soglia (rosso) in asenza di guasti sugli attuatori.

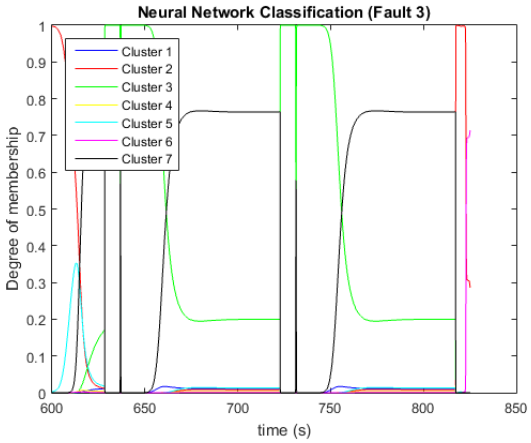
thruster.
A titolo dimostrativo riportiamo i grafici in figura 5.12, nei quali sono presentate le uscite del classificatore neurale a partire dall'istante in cui il guasto è introdotto nel sistema ($t = 600\ [s]$) fino al termine della simulazione, interrotta nel momento in cui il segnale d'allarme viene generato.
Possiamo subito osservare come nessuna simulazione sia stata eseguita in modo completo (1900 $[s]$) ma è stata sempre interrotta proprio a causa della rilevazione di un guasto. Indipendentemente dal thruster danneggiato, perciò, la PCA è in grado di risolvere il problema di FD. I tempi impiegati dall'algoritmo per individuare l'anomalia sono all'incirca gli stessi osservati per la survey descritta nel paragrafo 3.1, fanno eccezione solamente la rottura del terzo e del settimo attuatore. Osservando il grafico in figura 5.10, infatti, ci si aspetterebbe che la PCA rilevi entrambi i guasti all'istante $t \approx 770\ [s]$, ovvero quando il veicolo inizia ad effettuare la prima curva dalla rottura del motore. In realtà ciò non succede, dal momento che la rottura del terzo motore viene rilevata molto in ritardo rispetto a quanto atteso ($t \approx 820\ [s]$) e quella del settimo, invece, viene rilevata con un grande anticipo ($t \approx 630\ [s]$).
Anche per quanto riguarda l'isolamento mediante classificatore neurale i risultati sono soddisfacenti per tutti i guasti eccetto quello sul terzo attuatore. Quest'ultimo, infatti non viene riconosciuto



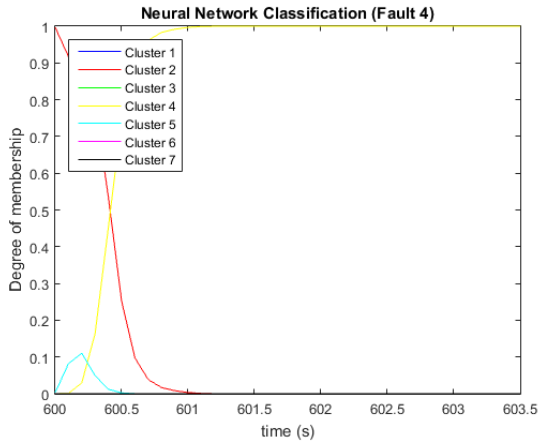
(a) *Fault 1.*



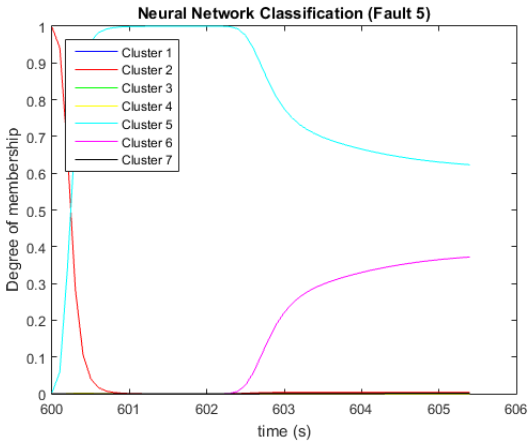
(b) *Fault 2*.



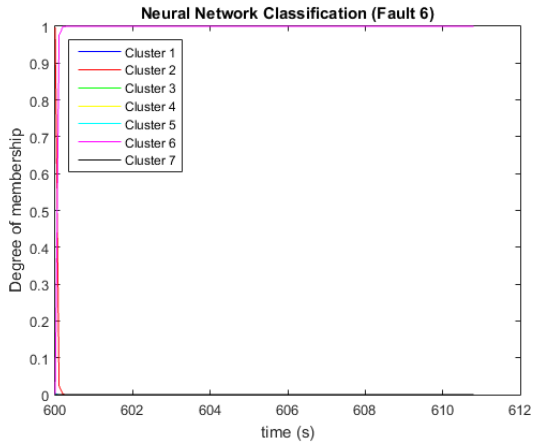
(c) *Fault 3*.



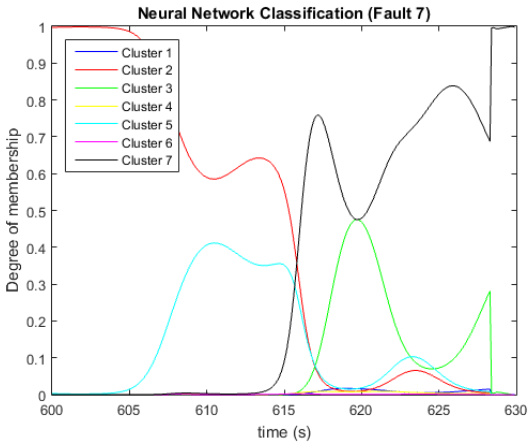
(d) *Fault 4*.



(e) *Fault 5*.



(f) *Fault 6*.



(g) *Fault 7*.

Figura 5.12: Uscita della rete neurale per FI durante l'intervallo temporale che va dall'insorgere del guasto ($t = 600\ s$) fino alla rilevazione dello stesso tramite PCA.

dal classificatore dal momento che l'uscita corrispondente è la più elevata solo per dei brevi periodi di prima della generazione del segnale d'allarme. Questo è forse dovuto alla scarsa influenza del thruster sulla dinamica del veicolo durante la survey, che, infatti, potrebbe essere tranquillamente portata a termine anche senza il suo utilizzo e con un lievissimo degrado delle prestazioni.

In figura 5.13 sono riportati, al variare del guasto, gli andamenti dei residui r_i calcolati mediante NLPCA durante la simulazione della survey. Anche in questo caso i grafici mostrano solo l'intervallo temporale che va dall'insorgere del guasto ($t = 600\ [s]$) fino alla sua rilevazione.

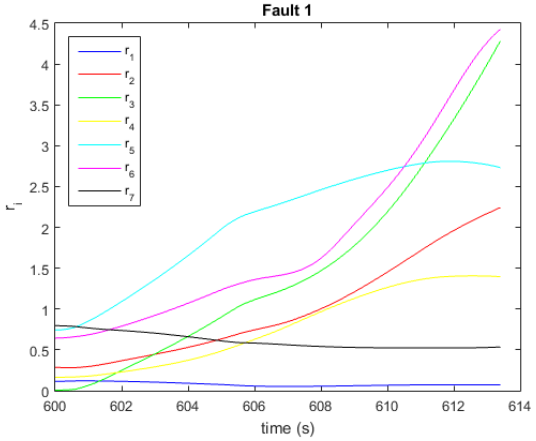
I risultati sono molto simili a quelli ottenuti con la tecnica della PCA e classificatore neurale, anche se è evidente un netto miglioramento per quanto riguarda rilevazione ed isolamento della rottura del thruster 3, che infatti stavolta viene identificata correttamente e con largo anticipo rispetto a quanto atteso. Nonostante questo, per quanto riguarda la parte di FD, potrebbero essere comunque preferibili le prestazioni della PCA, dal momento che riesce ad individuare i guasti sui thruster 4 e 5 in quasi la metà del tempo rispetto alla versione non lineare e, come abbiamo già visto nel paragrafo 5.1, tale differenza potrebbe ripercuotersi sulle prestazioni del sistema di fault recovery.

Finora abbiamo sempre supposto che il veicolo stesse svolgendo una survey oceanografica. Potrebbe essere interessante, però, studiare le prestazioni degli algoritmi di FD tramite PCA e NLPCA, quando il veicolo sta effettuando una missione diversa e quindi contenente delle manovre non presentate durante la fase di addestramento. Un esempio di questo tipo potrebbe essere una missione di sorveglianza di un perimetro noto, come l'ottagono mostrato in figura 5.14.

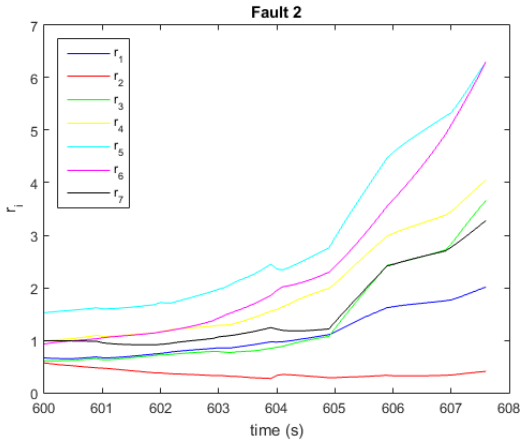
È evidente che il percorso desiderato non può essere scelto in modo casuale. Nel paragrafo 3.1, infatti, abbiamo già discusso su come sia irrealistico pretendere che gli algoritmi rispondano in modo corretto a manovre come il cambiamento di profondità o curve con rotazioni dell'angolo di imbardata maggiori di 90°. Nella traiettoria mostrata in figura 5.15, invece, il veicolo si limita sempre a seguire tratti rettilinei e a fare curve meno strette (rotazioni di circa 45° dell'angolo di imbardata) rispetto a quelle eseguite durante le survey (figure 3.2 e 5.10). Perciò si tratta di verificare se gli algoritmi riescono a individuare o meno la presenza di anomalie anche in corrispondenza di manovre differenti rispetto a quelle presentate durante la fase di addestramento ma pur sempre trattabili come una versione semplificata di queste.

In figura 5.16 sono mostrati gli andamenti dei residui corrispondenti a ciascuno dei due algoritmi studiati, durante la missione di sorveglianza e in assenza di guasti sugli attuatori. Vediamo che entrambi i segnali si mantengono sempre al di sotto della soglia di allarme, indicando che anche le curve a 45° sono state riconosciute correttamente come comportamenti normali e non come anomalie.

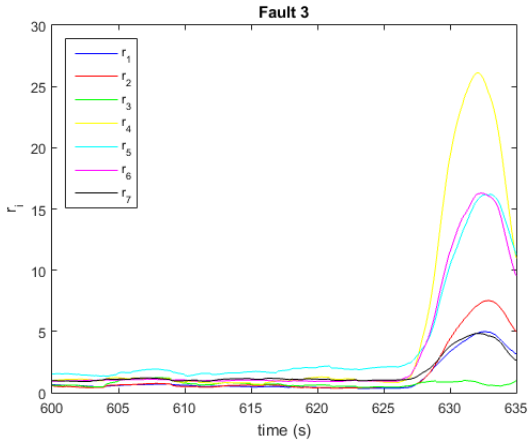
Nelle figure 5.17 e 5.18 sono riportate le uscite dei moduli di FD di entrambi gli algoritmi



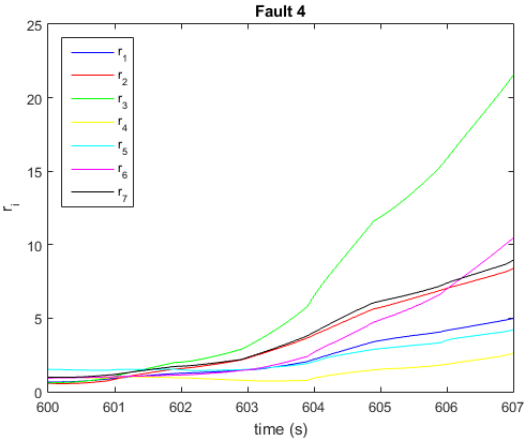
(a) *Fault 1*.



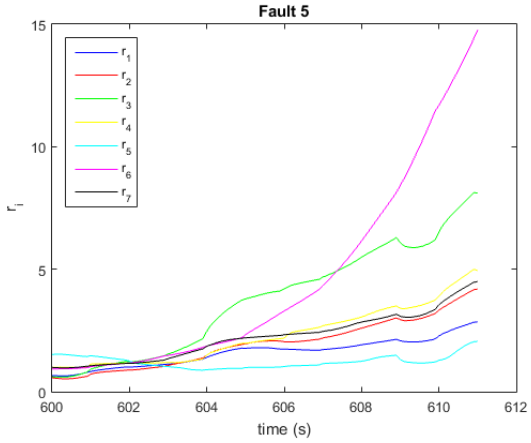
(b) *Fault 2*.



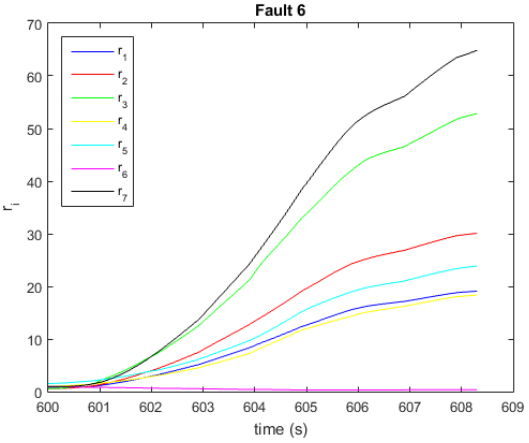
(c) *Fault 3*.



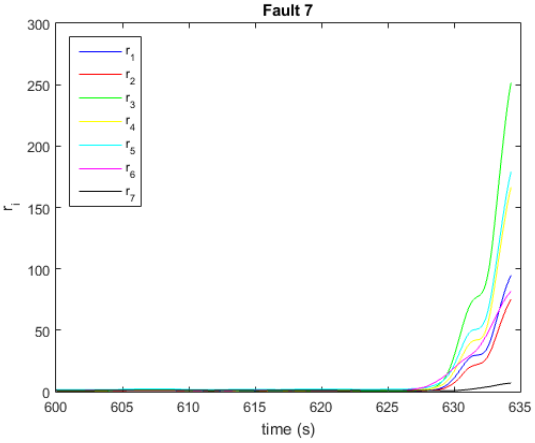
(d) *Fault 4*.



(e) *Fault 5*.



(f) *Fault 6*.



(g) *Fault 7*.

Figura 5.13: Residui calcolati con NLPCA durante l'intervallo temporale che va dall'insorgere del guasto ($t = 600\ s$) fino alla rilevazione dello stesso.

in corrispondenza rispettivamente di un guasto sul terzo e sul settimo thruster, simulato a partire dall'istante $t = 100\ [s]$.

Anche in questo caso i residui si mantengono sempre al di sotto della soglia di allarme non riuscendo quindi a riconoscere la rottura dei due motori. In realtà questo risultato non è particolarmente sorprendente, infatti si è già discusso ampliamente su come questi due attuatori abbiano

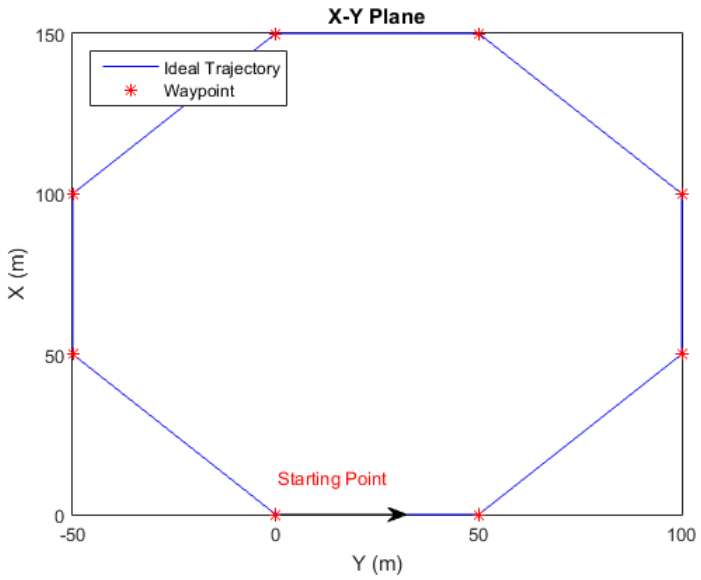


Figura 5.14: Waypoint (rosso) e traiettoria desiderata lungo il piano $X-Y$ (blu) per la missione di sorveglianza.

una scarsa influenza sul sistema e agiscano solo quando il veicolo deve virare. Nella missione di sorveglianza, però, le curve sono molto più morbide rispetto a quelle incontrate durante la survey, questo significa che il contributo fornito dai due attuatori risulta ancora più scarso.
In figura 5.19 sono riportate le traiettorie compiute dal veicolo sia in assenza di guasti sia in presenza di un guasto sul thruster 3 sia in presenza di un guasto sul thruster 7. Tali curve risultano praticamente sovrapposte nonostante a bordo non sia presente alcun sistema di fault recovery. Questo ci porta a concludere che il mancato riconoscimento di questi due guasti non compromette in alcun modo la funzionalità degli algoritmi di FD.

I grafici in figura 5.20 mostrano le uscite del classificatore neurale, utilizzato per l'isolamento

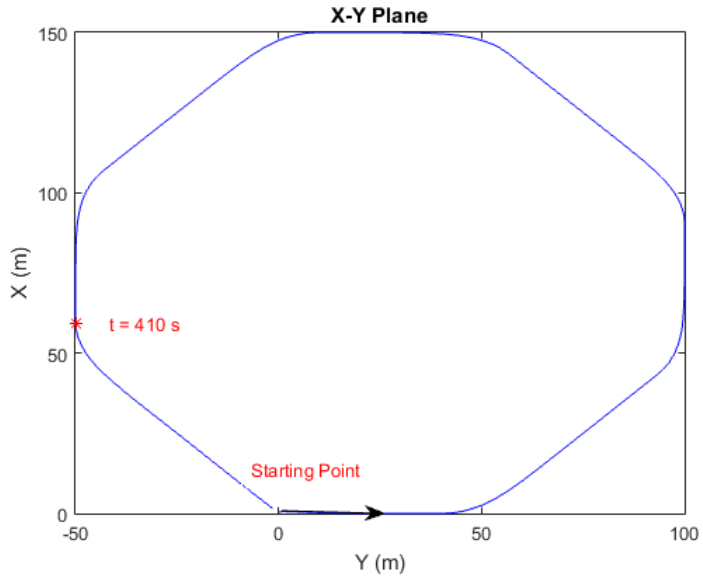
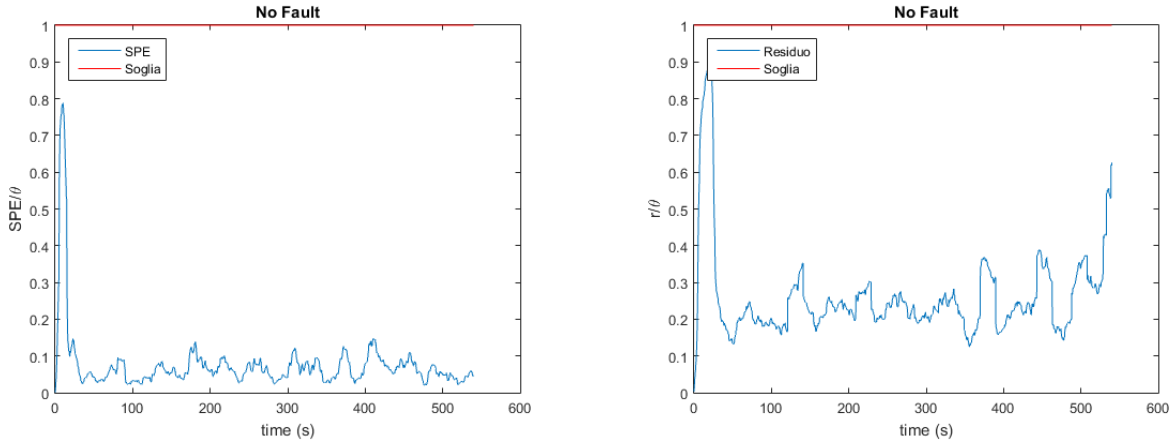


Figura 5.15: Traiettoria percorsa lungo il piano $X-Y$ (blu) in assenza di guasti, durante la missione di sorveglianza.



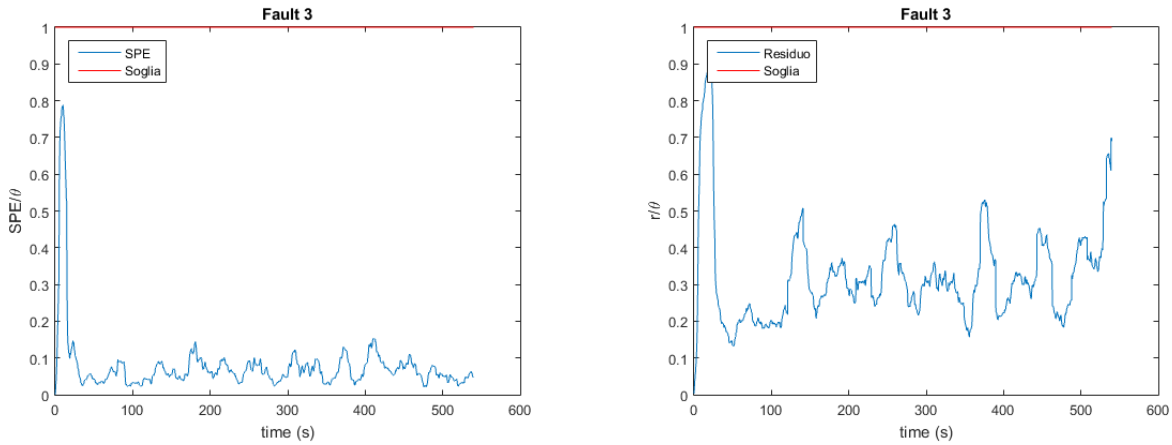
(a) *FD con PCA.* (b) *FD con NLPCA.*

Figura 5.16: Andamento nel tempo dei residui per Fault Detection (sia con PCA sia con NLPCA) (blu) e relativa soglia (rosso) in asenza di guasti sugli attuatori, durante la missione di sorveglianza.

dei guasti, durante l'intervallo di tempo che va dalla rottura del thruster ($t = 410\ [s]$) fino alla generazione del segnale di allarme da parte del modulo di FD tramite PCA. Per quanto detto prima, sono chiaramente esclusi i risultati relativi alla rottura degli attuatori 3 e 7.

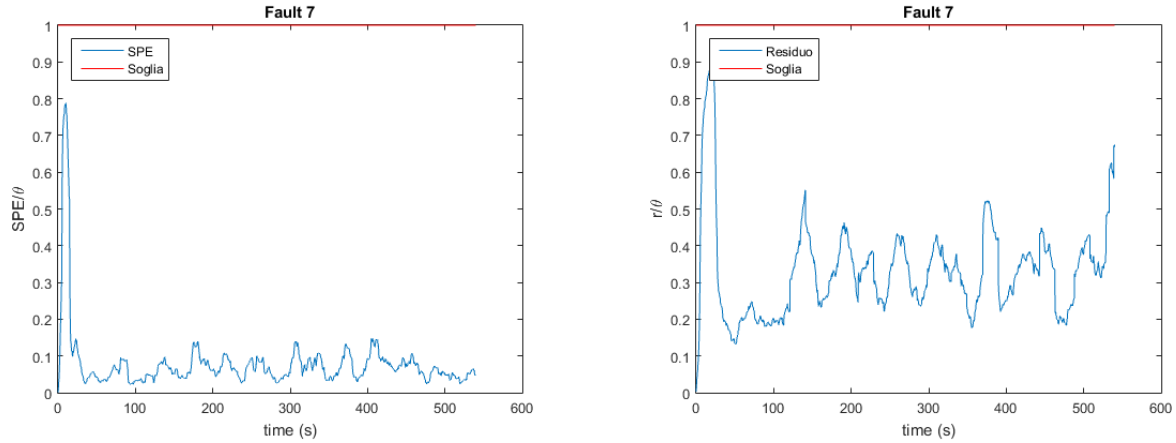
Osservando il grafico in figura 5.20c possiamo notare come il classificatore neurale suggerisca una rottura del sesto thruster quando, in realtà, a rompersi è stato il quarto. Perciò, nonostante la PCA sia in grado di rilevare i guasti già a pochi secondi dopo la loro comparsa, il classificatore neurale non risulta altrettanto affidabile nel fornire la sua risposta.

In figura 5.21, invece, sono riportati gli andamenti nel tempo dei residui associati ai 7 modelli NLPCA descritti nel paragrafo 4.6.Possiamo osservare come tutti i guasti siano stati rilevati ed isolati in modo esatto dopo un breve intervallo dalla rottura degli attuatori. Nonostante questo risultato sia positivo, è comunque doveroso notare che anche se l'isolamento è avvenuto correttamente, gli scarti fra i residui risultano molto piccoli. Rimane quindi il rischio che la presenza di



(a) *FD con PCA.* (b) *FD con NLPCA.*

Figura 5.17: Andamento nel tempo dei residui per Fault Detection (sia con PCA sia con NLPCA) (blu) e relativa soglia (rosso) in presenza di un guasto sul thruster 3 (simulato a partire dall'istante ($t = 100\ [s]$), durante la missione di sorveglianza.



(a) *FD con PCA.* (b) *FD con NLPCA.*

Figura 5.18: Andamento nel tempo dei residui per Fault Detection (sia con PCA sia con NLPCA) (blu) e relativa soglia (rosso) in presenza di un guasto sul thruster 7 (simulato a partire dall'istante ($t = 100$ $[s]$), durante la missione di sorveglianza.

dinamiche non modellate o disturbi non previsti, possano portare ad un'errata identificazione del guasto.

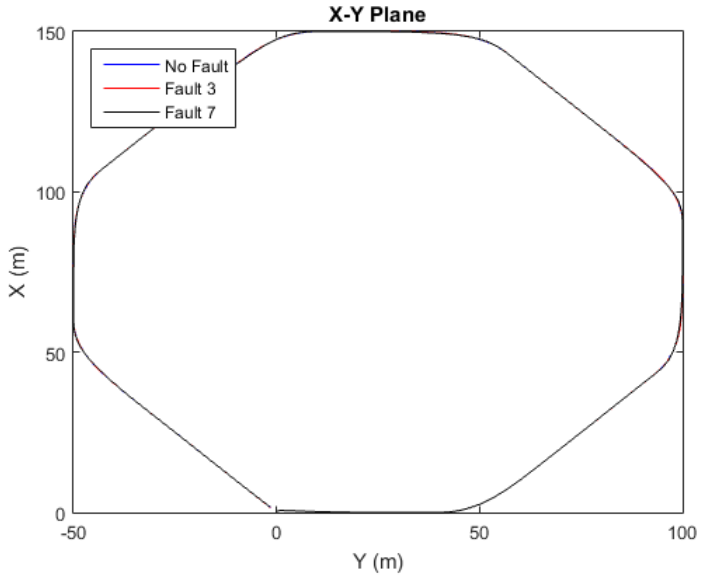
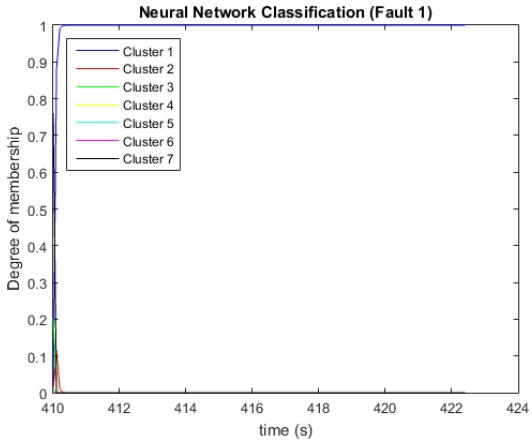
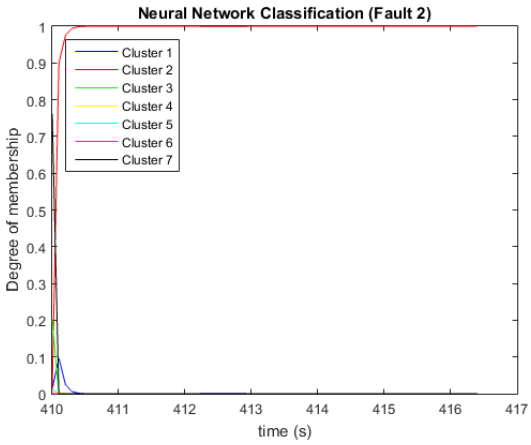


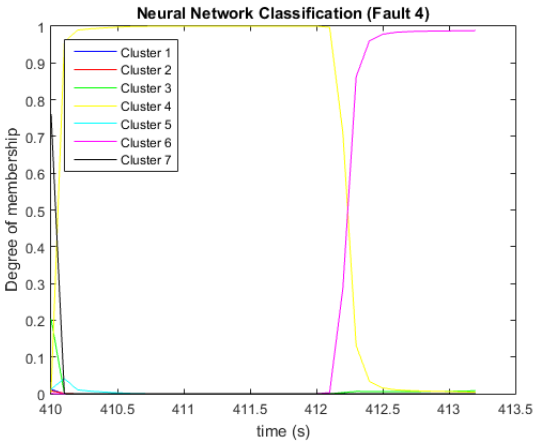
Figura 5.19: Traiettoria percorsa lungo il piano $X - Y$, durante la missione di sorveglianza, sia in assenza di guasti (blu),sia in presenza di un guasto sul thruster 3 (rosso), sia in presenza di un guasto sul thruster 7 (nero). Entrambi i guasti sono stati simulati a partire dall'istante $t = 100$ s



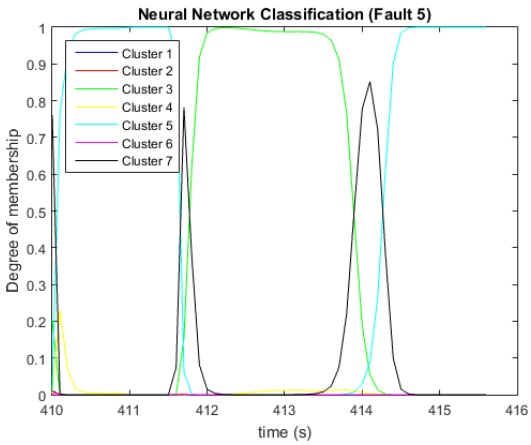
(a) *Fault 1*.



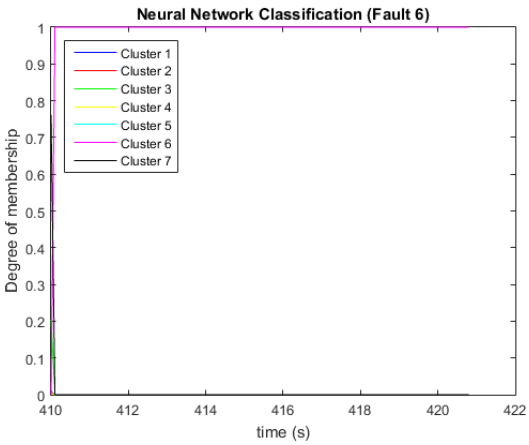
(b) *Fault 2*.



(c) *Fault 4*.

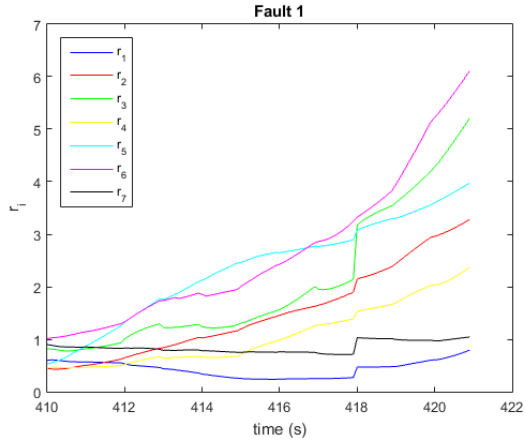


(d) *Fault 5*.

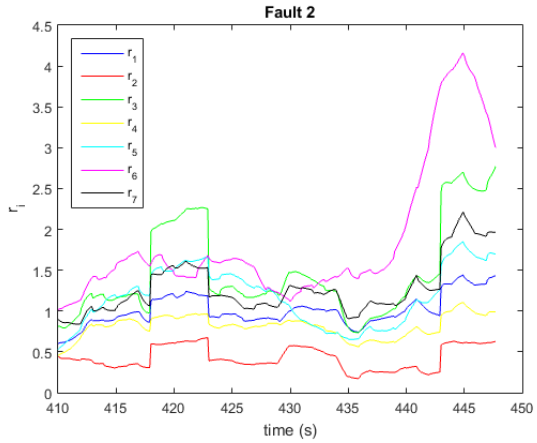


(e) *Fault 6*.

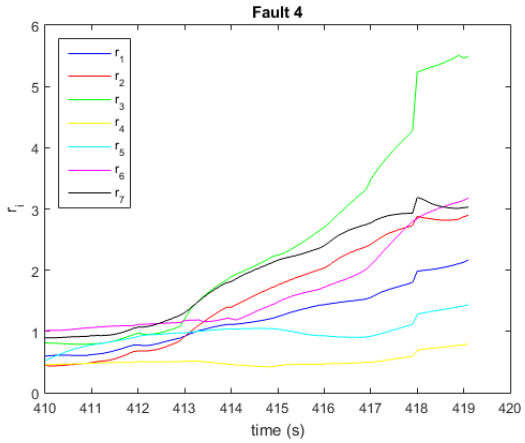
Figura 5.20: Uscita della rete neurale per FI durante l'intervallo temporale che va dall'insorgere del guasto ($t = 410\ s$) fino alla rilevazione dello stesso tramite PCA, durante la missione di sorveglianza.



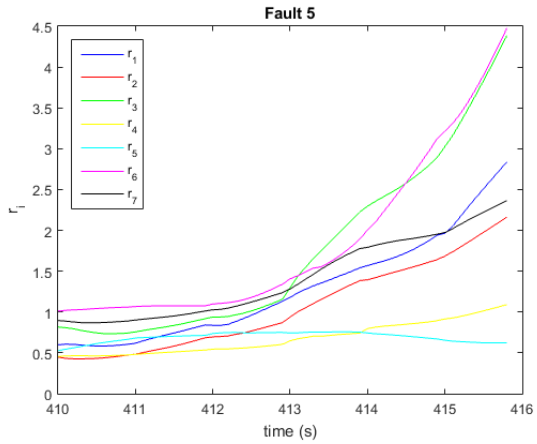
(a) *Fault 1*.



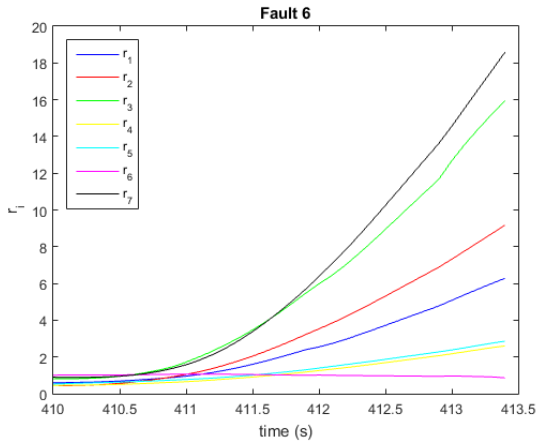
(b) *Fault 2*.



(c) *Fault 4*.



(d) *Fault 5*.



(e) *Fault 6*.

Figura 5.21: Residui calcolati con NLPCA durante l'intervallo temporale che va dall'insorgere del guasto ($t = 410\ s$) fino alla rilevazione dello stesso, durante la missione di sorveglianza.

CAPITOLO 6

CONCLUSIONI

Durante lo svolgimento della tesi le tecniche dell'Analisi delle Componenti Principali (PCA) e la sua variante non lineare (NLPCA), tipicamente utilizzate per la risoluzione del problema di fault detection nell'ambito degli impianti chimici, sono state studiate e riadattate per essere applicate su un veicolo subacqueo sovra-attuato.
I vantaggi offerti da queste due strategie sono notevoli, infatti la loro indipendenza dal modello matematico utilizzato garantisce una certa robustezza agli algoritmi sviluppati, che non risentono, perciò, delle incertezze sui parametri dinamici. D'altra parte è stato messo in evidenza come l'indipendenza dal modello abbia anche dei contro. L'algortimo, infatti, non è in grado di rispondere correttamente ad una situazione non prevista dal dataset di addestramento che quindi deve essere scelto con molta cura.
Nonostante entrambi gli algoritmi riescano sempre a rilevare i guasti prima che questi portino il veicolo all'instabilità, nessuno dei due è in grado di fornire una strategia di isolamento soddisfacente. Entrambi, infatti, richiedono che ogni guasto sia associato ad un'unica variabile, ma mentre un'ipotesi di questo tipo è adatta nel trattare guasti sui sensori o su attuatori di sistemi lenti e fortemente disaccoppiati, per un veicolo come V-Fides risulta completamente inadeguata.
Per risolvere questo problema si è deciso di elaborare un modello PCA e NLPCA del sistema per ognuno dei possibili guasti e di isolare il guasto osservando quale dei modelli elaborati descrive meglio le misure acquisite. La soluzione adottata, purtroppo, presenta un duplice svantaggio. Il primo è dovuto alla scelta del dataset di addestramento che, se non sufficientemente ricco, può portare ad un errato isolamento del guasto. Il secondo invece riguarda la dipendenza dell'algoritmo dal modello matematico del veicolo: mentre il dataset in condizioni operative può essere ottenuto direttamente da misure sul campo, quelli utilizzati per addestrare i modelli in presenza di thruster danneggiati devono essere necessariamente acquisiti in simulazione. Ad ogni modo i test effettuati mostrano che se la modellazione è fatta mediante componenti principali non lineari, allora l'algoritmo presenta una buona robustezza ai cambiamenti di parametri dinamici.
Dal confronto fra le velocità di rilevazione dei guasti della PCA e della NLPCA, si è potuto osservare come la prima fornisca risultati migliori in tutte le situazioni in cui la presenza di un sistema di fault recovery possa effettivamente migliorare le prestazioni del veicolo. Per questo motivo una soluzione interessante potrebbe essere quella di combinare le due tecniche utilizzando la PCA per risolvere il problema di fault detection e la NLPCA per quello di fault isolation.
Sviluppi futuri potrebbero riguardare la realizzazione di metodi alternativi per l'isolamento dei guasti da combinare con la PCA, oppure l'elaborazione di più modelli PCA (o NLPCA) che descrivano manovre specifiche che finora non sono state considerate, come il cambiamento di profondità da parte del veicolo. Ogni modello, selezionato da un sistema di controllo specifico basato sulla traiettoria che il veicolo sta compiendo, potrebbe essere in grado di fornire una risposta più affidabile da momento che verrebbe addestrato con un dataset più specifico.

BIBLIOGRAFIA

[1] Andrea Caiti, Francesco Di Corato et al. «The project V-fides: A new generation AUV for deep underwater exploration, operation and monitoring». In: *Oceans-St. John's, 2014, IEEE* (2014), pp. 1–7.

[2] Hui Cheng, Mats Nikus e Sirkka-Liisa Jämsä-Jounela. «Evaluation of PCA methods with improved fault isolation capabilities on a paper machine simulator». In: *Chemometrics and Intelligent Laboratory Systems* vol. 92.no. 2 (2008), pp. 186–199.

[3] Dong Dong e Thomas J. McAvoy. «Nonlinear principal component analysis - based on principal curves and neural networks». In: *American Control Conference* vol. 2 (1994), pp. 1284–1288.

[4] Ricardo Dunia e S. Joe Qin. «Subspace Approach to Multidimensional Identification and Reconstruction Fault». In: *American Institute of Chemical Engineers Journal* vol. 44.no. 8 (1998), pp. 1813–1831.

[5] Filippo Fabiani. «Rilevazione ed isolamento di guasti su attuatori di un veicolo marino sovra-attuato». Ingegneria Robotica e dell'Automazione. Università di Pisa, 2015.

[6] Thor I Fossen. «Guidance and control of ocean vehicles». In: *Wiley* vol. 199 (1994), pp. 1813–1831.

[7] Trevor Hastie e Werner Stuetzle. «Principal curves». In: *Journal of the American Statistical Association* vol. 84.no. 406 (1989), pp. 502–516.

[8] Kur Hornik, Maxwell Stinchcombe e Halber White. «Multilayer feedforward neural networks are universal approximators». In: *Neural Networks* vol. 2 (1989), pp. 359–366.

[9] Yunbing Huang, Janos Gertler e Thomas J. McAvoy. «Sensor and actuator fault isolation by structured partial PCA with nonlinear extensions». In: *Neural Networks* vol. 10 (2000), pp. 459–469.

[10] Yvon Tharrault, Gilles Mourot et al. «Fault detection and isolation with robust principal component analysis». In: *International Journal of Applied Mathematics and Computer Science* vol. 18.no. 4 (2008), pp. 429–442.

[11] Sergio Valle, Weihua Li e S. Joe Qin. «Selection of the Number of Principal Components: The Variance of the Reconstruction Error Criterion with a Comparison to Other Methods». In: *Industrial Engineering & Chemistry Research* vol. 38.no. 11 (1999), pp. 4389–4401.

[12] Silvia M. Zanoli e Giacomo Astolfi. «Application of a Fault Detection and Isolation System on a Rotary Machine». In: *International Journal of Rotating Machinery* vol. 2013 (2013).

[13] Silvia M. Zanoli e Giacomo Astolfi. «Application of a Mahalanobis-based Pattern Recognition technique for Fault Diagnosis on a chemical process». In: *Control & Automation (MED), 2012 20th Mediterranean Conference on*. IEEE. 2012, pp. 1347–1352.

[14] Silvia M. Zanoli, Giacomo Astolfi e Luca Barboni. «Applications of fault diagnosis techniques for a multishaft centrifugal compressor». In: *Control & Automation (MED), 2010 18th Mediterranean Conference on*. IEEE. 2010, pp. 64–69.

[15] Silvia M. Zanoli, Giacomo Astolfi e Luca Barboni. «FDI of Process Faults based on PCA and Cluster Analysis». In: *Control and Fault-Tolerant Systems (SysTol), 2010 Conference on*. IEEE. 2010, pp. 197–202.